
THE UNIVERSITY *of* EDINBURGH

# WALSH FUNCTIONS : SHAPE ANALYSIS AND OTHER APPLICATIONS

N. H. Searle


Ph.D.

University of Edinburgh

1970

ACKNOWLEDGEMENTS

This work was undertaken at the suggestion of Dr B. Meltzer of the Metamathematics Unit of the University of Edinburgh. Dr Meltzer has been a constant source of encouragement throughout the period of study.

The ever increasing number of workers in the field of Walsh functions has been an inspiration, and I am grateful to them all.

CONTENTS

## ABSTRACT

Due to their binary nature, the Walsh functions have considerable advantages over the traditional sinusoidal functions used in Fourier analysis when the computations are carried out by a general purpose binary digital computer. The important properties of the Walsh functions which illustrate these advantages are examined and developed. The Walsh transform and spectrum are presented in relation to the problem of function approximation, and various computational procedures for effecting the transform are explained.

The unconventional 'logical' transform is developed from the Walsh transform, and there is a discussion on the subject of interpreting the resulting spectrum.

There are other functions, such as the Haar functions, which are closely related to the Walsh functions, and their advantages are indicated.

The process of shape analysis is dealt with in terms of its relation to the more widely treated problem of pattern recognition. An application of shape analysis, using Walsh functions, to a study of leaf shapes is illustrated by experimental results.

A completely different approach to shape analysis is taken in the chapter on Pattern Generation and Simulation of Growth Processes. Other applications of Walsh functions, particularly of the 'logical' transform, are discussed in the final chapter.

Throughout, tested computer programs are used to provide examples, back up conjectures, and generally illustrate numerous points in the text.

# CHAPTER 1

## INTRODUCTION

A large amount of research effort has been, and is being, invested in the development of pattern recognition systems. The process of pattern recognition using a digital computer and appropriate peripherals is a multi-stage one. Firstly, the pattern must be input to the computer and possibly some picture processing procedures, such as de-blurring, are carried out. Then the data which has been input is transformed into a description of the pattern. The description may be numerical or linguistic, although even in the latter case it will of necessity be coded in numerical form, with a linguistic interpretation. Finally, on the basis of the resulting description, the pattern is classified, or recognised. The central element of the whole process from input to categorization is the construction of a suitable description.

It is desirable that the description should contain considerably less information than the original input data, but at the same time that it should form an adequate basis upon which recogntion can take place.

The input data, at one extreme, and the classification assigned to a patterm, at the other, are both descriptions of the pattern. Pattern recognition is the process by which one is transformed into the other.

There are, however, some applications in which it is not required that the entire process be carried out, but simply that a description of the type produced by the second stage of a pattern recognition system - one which forms a basis for classification - is constructed. This problem of pattern description, rather than pattern recognition, has been greatly neglected.

> "Categorization, clearly, is only one aspect of the recognition problem; not the whole of it by any means. It is our contention that the aim of any recognition procedure should not be merely to arrive at a 'Yes', 'No', 'Don't know' decision but to produce a structured description of the input picture. Perhaps a good part of this confusion about aims might have been avoided if, historically, the problem had been posed as not one of pattern recognition but of pattern analysis and description." (20).

A particular instance of an area of research where pattern description rather than pattern recognition is required is the botanist's study of leaf shapes. Trees and plants can indeed be identified by the shape of their leaves, but of greater interest to the botanist is the development of the leaf shape as the plant undergoes various stages of growth. By studying the changes of leaf shape the botanist hopes to gain some insight into the mechanisms of plant growth.

A technique for numerically describing, or specifying, shape will enable the botanist to quantify changes in shape, and to hypothesise functional relationships between leaves at different stages of the growth process. Such a technique, treating the leaf as a two-dimensional function and using the Walsh functions as approximators to that function, will be presented in this thesis, together with experimental results.

The hardware and software requirements for the implementation of a proper program of research into shape description are discussed.

A completely different approach to shape description, involving methods of growth simulation based on those of D. Cohen (5) and M. Eden (8,9), is investigated. The models presented in this thesis mainly serve to illustrate a discussion of the overall problem rather than constitute a solution to it.

The Walsh functions are particularly suited to computation using a binary digital computer because of their own essentially binary nature - they take only the values 1 and -1. Their use in function approximation parallels that of the sine and cosine functions in Fourier analysis. The computational advantages of the Walsh functions are exploited by means of a number of algorithms, including the Fast Walsh Transform, a direct analogue of the better-known Fast Fourier Transform. Exploiting the simple binary structure of the Walsh functions still further, an entirely new, non-arithmetic transform - the logical transform - is presented, and its applications discussed.

Chapter 2 defines and describes the Walsh functions, and exhibits some of the properties which are utilized in later chapters. The concept of pattern products is new, as are the statements and proofs of many of the theorems.

The computational procedures of Chapter 3 indicate the superiority of the Walsh functions over the sinusoidal functions, which is being increasingly recognised. The fast Walsh transform was discovered by the author in 1967. Its origin is uncertain. Because of its similarity to the fast Fourier transform, it is possible that it had previously been overlooked or taken for granted as a special case. The modification to the transform which allows for parallel computation was known for the fast Fourier transform, but had not before been applied to the fast Walsh transform.

The logical transform of Chapter 4, discovered by the author in 1968, is an innovation which represents the natural conclusion of the efforts to minimise the computational work involved in function approximation. It also leads to a rethinking of the interpretation of transform spectra.

From a computational point of view the Haar functions are possibly even better suited to shape description than the Walsh functions. However, the Haar transform does not give rise to a spectrum with the ideal interpretative properties of the Walsh spectrum.

Chapter 6 discusses the analysis of shapes and the part which Walsh functions might have to play in the process. The work described here was the first to apply Walsh functions to the analysis of two-dimensional pictures, and has been followed by many others, (2, 28).

The contrast between the methods of Chapter 6 and those of others who have used orthonormal functions in the solution of pattern recognition-type problems is dealt with in Chapter 7.

Subsequent chapters deal with numerical specification of leaf shapes, hardware and software for picture processing, growth simulation and automatic pattern generation, as well as other applications of Walsh functions and of the logical transform.

# CHAPTER 2

## WALSH FUNCTIONS : DEFINITIONS AND PROPERTIES

The Walsh functions are a complete set of orthonormal functions. They are both simple and rich in structure.

A number of definitions and theorems are presented below. They are intended to be neither comprehensive nor representative; they illustrate those particular properties which will be utilized in the applications which follow in later chapters.

### Walsh's definition

The Walsh functions are denoted by $w_0(x)$, $w_1(x)$, . . . and are defined for values of $x$ between 0 and 1.

$$w_0(x) = 1 \qquad 0 \leqslant x < 1$$

$$w_1(x) = \begin{cases} 1 & 0 \leqslant x < \frac{1}{2} \\ -1 & \frac{1}{2} \leqslant x < 1 \end{cases}$$

$$w_2^1(x) = \begin{cases} 1 & 0 \leqslant x < \frac{1}{4},\ \frac{3}{4} \leqslant x < 1 \\ -1 & \frac{1}{4} \leqslant x < \frac{3}{4} \end{cases}$$

$$w_2^2(x) = \begin{cases} 1 & 0 \leqslant x < \frac{1}{4},\ \frac{1}{2} \leqslant x < \frac{3}{4} \\ -1 & \frac{1}{4} \leqslant x < \frac{1}{2},\ \frac{3}{4} \leqslant x < 1 \end{cases}$$

and in general:

$$w_{n+1}^{2k-1}(x) = \begin{cases} w_n^k(2x) & 0 \leqslant x < \frac{1}{2} \\ (-1)^{k+1} w_n^k(2x-1) & \frac{1}{2} \leqslant x < 1 \end{cases}$$

$$w_{n+1}^{2k}(x) = \begin{cases} w_n^k(2x) & 0 \leqslant x < \frac{1}{2} \\ (-1)^{k} w_n^k(2x-1) & \frac{1}{2} \leqslant x < 1 \end{cases}$$

$$\text{for} \quad k = 1, 2, \ldots, 2^{n-1}$$
$$n = 1, 2, \ldots$$

(29)

At points of discontinuity, the function takes the value of the average of the two limits approached on the two sides of the discontinuity.

Whilst the above definition is independent of the Haar functions, Walsh showed that his functions could each be expressed as a linear combination of a finite number of the Haar functions. At the same time, there are very considerable differences between the two sets of functions with respect to their essential properties. The Walsh functions, at least superficially, are strikingly similar to the sine and cosine functions (11), whilst the Haar functions are most dissimilar.

The Haar functions are discussed in detail in Chapter 5, where it will be shown that they have a computational advantage over the Walsh functions. However, the latter possess three important properties which the former do not:

(i) The nth Walsh function has $n-1$ sign changes in the interval considered.

(ii) Each Walsh function takes one of only two values, +1 or -1, everywhere in the interval.

(iii) Each Walsh function is either odd or even with respect to the midpoint of the interval.

An alternative definition

By referring to the "nth Walsh function" in (i) above, it is assumed that the ordering of the Walsh functions is as follows:

$$w_0 \quad w_1 \quad (w_2^1 \quad w_2^2) \quad (w_3^1 \quad w_3^2 \quad w_3^3 \quad w_3^4) \ldots$$

$$(w_k^1 \quad w_k^2 \ldots w_k^{2^{n-1}}) \ldots$$

This is expressed formally by saying that if the binary expansion of $m$ is $d_1 d_2 \ldots d_n$, where $d_1$ is unity, and $k$ is the value of $d_2 d_3 \ldots d_n$, then

$$w_m(x) = w_n^{k+1}(x)$$

A concise definition, consistent with the above single-index notation, is as follows:

$$w_0(x) = 1 \qquad 0 \leqslant x < 1$$

$$w_k(x) = \begin{cases} w_j(2x) & 0 \leqslant x < \frac{1}{2} \\ (-1)^{k+j} w_j(2x-1) & \frac{1}{2} \leqslant x < 1 \end{cases}$$

where $j$ is the integer part of $k/2$

The integer part notation is a little clumsy, although readily computable, and the definition may be re-written as:

$$w_0(x) = 1 \qquad 0 \leqslant x < 1$$

$$w_{2k}(x) = \begin{cases} w_k(2x) & 0 \leqslant x < \frac{1}{2} \\ (-1)^k w_k(2x-1) & \frac{1}{2} \leqslant x < 1 \end{cases}$$

$$w_{2k+1}(x) = \begin{cases} w_k(2x) & 0 \leqslant x < \frac{1}{2} \\ (-1)^{k+1} w_k(2x-1) & \frac{1}{2} \leqslant x < 1 \end{cases}$$

All these definitions of the Walsh functions can be extended from the interval from 0 to 1 to any part, or the whole, of the real line. Also, the Walsh functions are simply extended to n dimensions.

Example:

$$w_{i,j}(x,y) = w_i(x).w_j(y)$$

The ordering of the Walsh functions in which the nth function has n-1 sign changes in the interval is denoted by W - the W ordering.

From this definition, we see that Walsh functions are derived from others by 'packing' a Walsh function lower in the W ordering into the first half of the interval, and repeating the procedure in the second half of the interval either with the same function or with the negation of it:

(In the table below, the indices of the Walsh functions are written in binary expanded form on the left hand side to illustrate the validity of the general case. $d_n$ and $d_{n-1}$ are the last two bits of the binary expansion of k and $\oplus$ is binary addition, modulo 2).

$$
\begin{aligned}
w_{000}(x) &= w_0(x) \qquad w_0(x) \\
w_{001}(x) &= w_0(x) \qquad -w_0(x) \\
w_{010}(x) &= w_1(x) \qquad -w_1(x) \\
w_{011}(x) &= w_1(x) \qquad w_1(x) \\
w_{100}(x) &= w_2(x) \qquad w_2(x) \\
w_{101}(x) &= w_2(x) \qquad -w_2(x) \\
w_{110}(x) &= w_3(x) \qquad -w_3(x) \\
w_{111}(x) &= w_3(x) \qquad w_3(x)
\end{aligned}
$$

$$
\vdots
$$

$$
w_k(x) = w_j(x) \qquad (-1)^{d_n \oplus d_{n-1}} w_j(x)
$$

$$
\vdots
$$

$j$ is the integer part of $k/2$.

Walsh matrices

It can be seen from the above definitions that each Walsh function is a step function, taking the value +1 or -1 in each of the $2^n$ subintervals of the interval from 0 to 1. Consequently, the functions may be adequately represented by patterns of +1s and -1s, simply lists of the $2^n$ values they take in the (equal) subintervals. For the function $w_k(x)$, $n$ is the smallest integer such that $k < 2^n$.

Example:

$$
\begin{aligned}
w_0(x) &= 1 \\
w_1(x) &= 1 \quad -1 \\
w_2(x) &= 1 \quad -1 \quad -1 \quad 1 \\
w_3(x) &= 1 \quad -1 \quad 1 \quad -1 \\
w_4(x) &= 1 \quad -1 \quad -1 \quad 1 \quad 1 \quad -1 \quad -1 \quad 1 \\
w_5(x) &= 1 \quad -1 \quad -1 \quad 1 \quad -1 \quad 1 \quad 1 \quad -1 \\
w_6(x) &= 1 \quad -1 \quad 1 \quad -1 \quad -1 \quad 1 \quad -1 \quad 1 \\
w_7(x) &= 1 \quad -1 \quad 1 \quad -1 \quad 1 \quad -1 \quad 1 \quad -1
\end{aligned}
$$

Note that the pattern 1 1 -1 -1, say, represents the same function ($w_1(x)$ in this case) as the pattern 1 -1 . The difference is simply due to whether we consider $w_1(x)$ to be one of the first four Walsh functions, or as one of the first two.

If the patterns corresponding to the first $2^n$ Walsh functions for some positive integer value of $n$ are each expanded so that they are of length $2^n$ values, then the Walsh matrix, $\underline{W}_n$ , can be formed.

$$
\underline{W}_3 = \begin{bmatrix}
+1 & +1 & +1 & +1 & +1 & +1 & +1 & +1 \\
+1 & +1 & +1 & +1 & -1 & -1 & -1 & -1 \\
+1 & +1 & -1 & -1 & -1 & -1 & +1 & +1 \\
+1 & +1 & -1 & -1 & +1 & +1 & -1 & -1 \\
+1 & -1 & -1 & +1 & +1 & -1 & -1 & +1 \\
+1 & -1 & -1 & +1 & -1 & +1 & +1 & -1 \\
+1 & -1 & +1 & -1 & -1 & +1 & -1 & +1 \\
+1 & -1 & +1 & -1 & +1 & -1 & +1 & -1
\end{bmatrix}
$$

The rows of the above matrix are the values of the first eight $(2^3)$ Walsh functions.

It will be proved that $\underline{W}_n$ is symmetrical for all non-negative integers n.

Rademacher functions

The Rademacher functions are a set of orthonormal functions which are not complete. They only take the values +1 and -1, and they are a proper subset of the Walsh functions. They are denoted by $R_0(x), R_1(x), \ldots$

$$R_0(x) = 1 \qquad 0 \leqslant x < 1$$

$$R_n(x) = \begin{cases} 1 & \frac{2k}{2^n} \leqslant x < \frac{2k+1}{2^n} \\ -1 & \frac{2k+1}{2^n} \leqslant x < \frac{2k+2}{2^n} \end{cases}$$

$$\text{for} \quad k = 0, 1, \ldots, 2^{n-1}-1$$

$$n = 1, 2, \ldots$$

$R_n(x)$ is a step function which takes the values +1 and -1 alternately in the $2^n$ equal subintervals of the interval from 0 to 1.

In fact,

$$R_n(x) = w_{2^n-1}(x).$$

However, we are concerned here with the definition of the Walsh functions in terms of the Rademacher functions rather than vice versa.

The Walsh functions are defined as all possible products of the Rademacher functions.

No Rademacher function appears more than once in a given product, since $R_m(x).R_m(x) = 1$ everywhere for all values of $m$.

This definition of the Walsh functions gives rise to a new ordering of them, which will be called the R ordering. The nth function in the R ordering will be denoted by $w'_n(x)$.

$$w'_0(x) = R_0(x)$$

and $$w'_{n_1 n_2 \ldots n_m}(x) = R_{n_1}(x).R_{n_2}(x) \ldots R_{n_m}(x)$$

where $$0 \leqslant n_1 < n_2 < \ldots < n_m$$

and $$w'_{n_1 n_2 \ldots n_m}(x) = w'_k(x)$$

where $k$ is the number which has unity in the $n_1$th, $n_2$th, . . . , $n_m$th places after the binary point in its binary expansion and zero elsewhere.

Example:

$$w'_5(x) = w'_{101}(x) = w'_{1.3}(x) = R_1(x).R_3(x)$$

Whilst the W ordering has important interpretative properties, the R ordering is suggestive of methods of computation which have great advantages over more conventional methods. The relationship between the two orderings will be more closely investigated later.

Pattern products

Suppose p is a pattern of numbers, in particular a list of +1s and -1s representing the values of a Walsh function, as in a row of a Walsh matrix.

Then the basic patterns, $\overline{0}$ and $\overline{1}$, are defined as follows:

$$\overline{0} = 1\ 1$$
$$\overline{1} = 1\ -1$$

Pattern products, products of the basic patterns, are defined:

$\overline{0}.p = p\ p$ i.e. the pattern p followed by a repetition of itself.

$\overline{1}.p = p\ -p$ i.e. the pattern p followed by the pattern in which each element is the negation of the corresponding element of p.

Example:

$$\overline{1} = 1\ -1$$
$$\overline{0}.\overline{1} = 1\ -1\ 1\ -1$$
$$\overline{1}.\overline{0}.\overline{1} = 1\ -1\ 1\ -1\ -1\ 1\ -1\ 1$$

Each of the first $2^n$ Walsh functions can be expressed as a product of not more than n factors, each of which is a basic pattern. In other words, a Walsh function is a basic pattern, or the pattern product of a Walsh function and a basic pattern.

The index of the Walsh function produced by a given pattern product is simply derived from the value of the binary number represented by the basic pattern factors.

Example:

$$\bar{1}.\bar{0}.\bar{1} \quad = \quad w'_{101}(x) \quad = \quad w'_{5}(x)$$

Note that the index is in the R ordering. Note also that the same Walsh function is represented by both $\bar{d}_1.\bar{d}_2. \ . \ . \ . \ \bar{d}_m.\bar{0}$ and $\bar{d}_1.\bar{d}_2. \ . \ . \ . \ \bar{d}_m$ . The concept of pattern products can be extended to n dimensions, and as such can be used to define n-dimensional Walsh functions.

The table which appears below shows both the W and the R ordering for the first $2^n$ Walsh functions, and in this case covers n = 0, 1, 2 and 3.

| W ordering | .000 | .001 | .010 | .011 | .100 | .101 | .110 | .111 | pp | R |
|---|---|---|---|---|---|---|---|---|---|---|
| 000 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | $\bar{0}.\bar{0}.\bar{0}$ | 0 |
| 001 | 1 | 1 | 1 | 1 | -1 | -1 | -1 | -1 | $\bar{1}.\bar{0}.\bar{0}$ | 4 |
| 010 | 1 | 1 | -1 | -1 | -1 | -1 | 1 | 1 | $\bar{1}.\bar{1}.\bar{0}$ | 6 |
| 011 | 1 | 1 | -1 | -1 | 1 | 1 | -1 | -1 | $\bar{0}.\bar{1}.\bar{0}$ | 2 |
| 100 | 1 | -1 | -1 | 1 | 1 | -1 | -1 | 1 | $\bar{0}.\bar{1}.\bar{1}$ | 3 |
| 101 | 1 | -1 | -1 | 1 | -1 | 1 | 1 | -1 | $\bar{1}.\bar{1}.\bar{1}$ | 7 |
| 110 | 1 | -1 | 1 | -1 | -1 | 1 | -1 | 1 | $\bar{1}.\bar{0}.\bar{1}$ | 5 |
| 111 | 1 | -1 | 1 | -1 | 1 | -1 | 1 | -1 | $\bar{0}.\bar{0}.\bar{1}$ | 1 |

pp : pattern product R : R ordering

The final column consists simply of the binary values of the penultimate column.

Modified Walsh matrices

A Walsh matrix, $\underline{W}_n$, has rows which represent the first $2^n$ Walsh functions in the W ordering. A modified Walsh matrix, $\underline{W}'_n$, has rows which represent the first $2^n$ Walsh functions in the R ordering.

A modified Walsh matrix is in fact a particular type of Hadamard matrix, and there is an extremely simple means of generating them:

$$\underline{W}'_0 = 1$$

$$\underline{W}'_n = \begin{bmatrix} \underline{W}'_{n-1} & \vdots & \underline{W}'_{n-1} \\ \cdots & \vdots & \cdots \\ \underline{W}'_{n-1} & \vdots & -\underline{W}'_{n-1} \end{bmatrix}$$

$$\text{for } n = 1, 2, \ldots \qquad (13)$$

The rows of $\underline{W}'_n$ are a permutation of the rows of $\underline{W}_n$, i.e. the first $2^n$ Walsh functions are the same set of functions in both the W and R orderings.

However, the kth row of $\underline{W}'_n$ is not in general equal to the kth row of $\underline{W}'_m$, where $m \neq n$. That is, the R ordering depends upon the value of n. This is not so in the W ordering, since there the kth function has k sign changes in the interval.

Example:

The first four functions in the R ordering in terms of the W ordering are:

$$w_0(x) \quad w_3(x) \quad w_1(x) \quad w_2(x)$$

The first eight functions in the R ordering in terms of the W ordering are:

$$w_0(x) \quad w_7(x) \quad w_3(x) \quad w_4(x) \quad w_1(x) \quad w_6(x) \quad w_2(x) \quad w_5(x)$$

Thus, as one of the first four functions, $w'_2(x)$ is $w_1(x)$ ; whereas, as one of the first eight, it is $w_3(x)$ .

Binary matrices

Suppose that the +1s and -1s of a modified Walsh matrix are replaced by 0s and 1s, respectively. Then the resultant matrix ( $2^n$ x $2^n$ ) is the product (modulo 2) of a $2^n$ x n matrix, whose rows are simply representations of the binary numbers in natural order from 0 to $2^n-1$, and its own transpose. A proof of an equivalent statement appears in the section of this chapter entitled 'Symmetry of Walsh matrices'.

Example:

$$\begin{bmatrix} 0&0&0 \\ 0&0&1 \\ 0&1&0 \\ 0&1&1 \\ 1&0&0 \\ 1&0&1 \\ 1&1&0 \\ 1&1&1 \end{bmatrix} \begin{bmatrix} 0&0&0&0&1&1&1&1 \\ 0&0&1&1&0&0&1&1 \\ 0&1&0&1&0&1&0&1 \end{bmatrix} = \begin{bmatrix} 0&0&0&0&0&0&0&0 \\ 0&1&0&1&0&1&0&1 \\ 0&0&1&1&0&0&1&1 \\ 0&1&1&0&0&1&1&0 \\ 0&0&0&0&1&1&1&1 \\ 0&1&0&1&1&0&1&0 \\ 0&0&1&1&1&1&0&0 \\ 0&1&1&0&1&0&0&1 \end{bmatrix}$$

When 0s are replaced by +1s and 1s by -1s , the right hand side is the modified Walsh matrix $\underline{W}'_3$ .

The binary Walsh matrix (unmodified) can be formed by the product (modulo 2) of two matrices, one of which consists simply of the binary numbers, as above, and the other of which consists of the binary numbers permuted in accordance with the relationship between the R and W orderings.

Example:

$$\begin{bmatrix} 0&0&0 \\ 1&0&0 \\ 1&1&0 \\ 0&1&0 \\ 0&1&1 \\ 1&1&1 \\ 1&0&1 \\ 0&0&1 \end{bmatrix} \begin{bmatrix} 0&0&0&0&1&1&1&1 \\ 0&0&1&1&0&0&1&1 \\ 0&1&0&1&0&1&0&1 \end{bmatrix} = \begin{bmatrix} 0&0&0&0&0&0&0&0 \\ 0&0&0&0&1&1&1&1 \\ 0&0&1&1&1&1&0&0 \\ 0&0&1&1&0&0&1&1 \\ 0&1&1&0&0&1&1&0 \\ 0&1&1&0&1&0&0&1 \\ 0&1&0&1&1&0&1&0 \\ 0&1&0&1&0&1&0&1 \end{bmatrix}$$

The right hand matrix is the binary Walsh matrix $\underline{W}_3$ .

The binary Walsh matrix can also be formed as the product (modulo 2 ) of a matrix, consisting of a permutation of the binary numbers, and its own transpose.

Example:

$$\begin{bmatrix} 0&0&0 \\ 0&1&1 \\ 1&0&0 \\ 1&1&1 \\ 1&0&1 \\ 1&1&0 \\ 0&0&1 \\ 0&1&0 \end{bmatrix} \begin{bmatrix} 0&0&1&1&1&1&0&0 \\ 0&1&0&1&0&1&0&1 \\ 0&1&0&1&1&0&1&0 \end{bmatrix} = \begin{bmatrix} 0&0&0&0&0&0&0&0 \\ 0&0&0&0&1&1&1&1 \\ 0&0&1&1&1&1&0&0 \\ 0&0&1&1&0&0&1&1 \\ 0&1&1&0&0&1&1&0 \\ 0&1&1&0&1&0&0&1 \\ 0&1&0&1&1&0&1&0 \\ 0&1&0&1&0&1&0&1 \end{bmatrix}$$

This permutation can be derived from the simpler one connecting the R and W orderings. The importance of the binary Walsh matrices is that they are the simplest form in which the values of the Walsh functions can be generated. Although this is not necessary in the fast Walsh transform, it is required for the 'logical' transform.

Relationship between R and W orderings

The relationship between the two orderings is the same as that between the normal binary code and the reflected binary code.

Note: in the following sections of this chapter, where the index of a Walsh function is written in its binary expansion form, commas are used to separate the bits of the binary expansion where necessary to avoid confusion.

Theorem

$$w^{!}_{d_1 d_2 \ldots d_m}(x) = w_{d_m, d_m \oplus d_{m-1}, \ldots, d_m \oplus d_{m-1} \oplus \ldots d_1}(x)$$

and

$$w_{d_1 d_2 \ldots d_m}(x) = w^{!}_{d_m \oplus d_{m-1}, d_{m-1} \oplus d_{m-2}, \ldots, d_2 \oplus d_1, d_1}(x)$$

where $0 \oplus 1 = 1 \oplus 0 = 1$

and $0 \oplus 0 = 1 \oplus 1 = 0$

i.e. $\oplus$ is addition (modulo 2) - no carry.

Proof

As a pattern product

$$w'_{d_1 d_2 \ldots d_m}(x) = \overline{d}_1 . \overline{d}_2 . \ldots . \overline{d}_m \qquad (I)$$

and

$$w_{d_1 d_2 \ldots d_m}(x) = \begin{cases} \overline{1} . (-1)^{d_m \oplus d_{m-1}} \, w_{d_1 d_2 \ldots d_{m-1}}(x) \\ \overline{0} . (-1)^{d_m \oplus d_{m-1}} \, w_{d_1 d_2 \ldots d_{m-1}}(x) \end{cases}$$

depending upon whether $d_m \oplus d_{m-1}$ is 1 or 0, from the definition on page 10.

$$= \overline{d_m \oplus d_{m-1}} \cdot w_{d_1 d_2 \ldots d_{m-1}}(x)$$

$$= \overline{d_m \oplus d_{m-1}} \cdot \overline{d_{m-1} \oplus d_{m-2}} \cdot \ldots \cdot \overline{d_2 \oplus d_1} \cdot w_{d_1}(x)$$

Now

$$w_{d_1}(x) = \begin{cases} 1 \quad 1 = \overline{0} & \text{if} \quad d_1 = 0 \\ 1 \quad -1 = \overline{1} & \text{if} \quad d_1 = 1 \end{cases}$$

So

$$w_{d_1 d_2 \ldots d_m}(x) = \overline{d_m \oplus d_{m-1}} \cdot \overline{d_{m-1} \oplus d_{m-2}} \cdot \ldots \cdot \overline{d_2 \oplus d_1} \cdot \overline{d_1}$$

$$= w'_{d_m \oplus d_{m-1}, d_{m-1} \oplus d_{m-2}, \ldots, d_2 \oplus d_1, d_1}(x)$$

from (I) above.

Symmetry of Walsh matrices

Theorem

$w_i^!(x)$ has the value of the scalar product (modulo 2) of the binary expansions of $i$ and $x$ .

This is equivalent to the statement that the modified binary matrix is the product (modulo 2) of the matrix of binary numbers in natural order and its own transpose. It is also equivalent to a theorem from Polyak and Shreider (24) which is given here with proof:

Theorem

$$w_i^!(x) = (-1)^{\langle i,x \rangle}$$

where $\langle i,x \rangle$ is the number of binary places in which both $i$ and $x$ have unity in their binary expansions.

Proof

This follows directly from:

(i) The definition of the Walsh functions in the R ordering in terms of the Rademacher functions.

(ii) The definition of the Rademacher functions, based on the observation that they are step-functions taking the values +1 and -1 alternately, viz:

$$R_n(x) = \begin{cases} 1 & \text{if the nth bit of } x \text{ is 0} \\ -1 & \text{if the nth bit of } x \text{ is 1} \end{cases}$$

(iii) The property of $\langle\ \rangle$

$$\langle a \oplus b, x\rangle = \langle a, x\rangle + \langle b, x\rangle \text{ (modulo 2)}$$

Theorem

The modified Walsh matrix, $\underline{W}'_n$, is symmetrical for all non-negative n.

Proof

Follows from the fact that $\langle i, x\rangle = \langle x, i\rangle$

Theorem

The Walsh matrix, $\underline{W}_n$, is symmetrical for all non-negative n.

Proof

From the definition of the Walsh functions given on page 10:

$$w_{d_1 d_2 \ldots d_m}(.r_1 r_2 \ldots r_m) = (-1)^{r_1 \wedge (d_m \oplus d_{m-1})}\, w_{d_1 d_2 \ldots d_{m-1}}(.r_2 r_3 \ldots r_m)$$

where

$$0 \oplus 0 = 1 \oplus 1 = 0 \quad \text{and} \quad 1 \oplus 0 = 0 \oplus 1 = 1$$

and $$1 \wedge 1 = 1 \quad \text{and} \quad 0 \wedge 0 = 1 \wedge 0 = 0 \wedge 1 = 0$$

If the reduction formula above is applied a further m times, the following result is obtained:

$$w_{d_1 d_2 \ldots d_m}(.r_1 r_2 \ldots r_m) = (-1)^{r_1 \wedge (d_m \oplus d_{m-1}) \oplus r_2 \wedge (d_{m-1} \oplus d_{m-2}) \oplus \ldots \oplus r_{m-1} \wedge (d_2 \oplus d_1)}\, w_{d_1}(.r_m) \qquad \text{(I)}$$

Note that $(-1)^a . (-1)^b = (-1)^{a \oplus b}$

Now, $w_0(0) = w_0(.1) = w_1(0) = 1$ and $w_1(.1) = -1$

and so

$$w_{d_1}(.r_m) = (-1)^{r_m \wedge d_1}$$

The right hand side of (I) becomes:

$$(-1)^{r_1 \wedge (d_m \oplus d_{m-1}) \oplus r_2 \wedge (d_{m-1} \oplus d_{m-2}) \oplus \ldots \oplus r_{m-1} \wedge (d_2 \oplus d_1) \oplus r_m \wedge d_1} \quad \text{(II)}$$

Since

$$a \wedge ( b \oplus c ) = a \wedge b \oplus a \wedge c \quad ,$$

the terms of the exponent can be regrouped to give:

$$(-1)^{d_m \wedge r_1 \oplus d_{m-1} \wedge (r_1 \oplus r_2) \oplus d_{m-2} \wedge (r_2 \oplus r_3) \oplus \ldots \oplus d_1 \wedge (r_{m-1} \oplus r_m)}$$

If the terms of the exponent are now written in reverse order and compared with the expression (II), the above expression is seen to be

$$w_{r_1 r_2 \ldots r_m}(.d_1 d_2 \ldots d_m)$$

which is the 'inverse' of the left hand side of (I).

Hence, the Walsh matrix is symmetrical for all non-negative values of the integer n .

Products of Walsh functions

Theorem

$$w'_i(x) . w'_j(x) = w'_{i \oplus j}(x)$$

Proof

If only either $w'_i(x)$ or $w'_j(x)$ contains the Rademacher function $R_k(x)$ as a factor in its definition as a product of Rademacher functions, then so will the product. If both $w'_i(x)$ and $w'_j(x)$ contain $R_k(x)$ as a factor, or if neither contain it, then the product will not.

The bits of i and j which are unity determine which Rademacher functions appear in the product. (see page 13).

As a special case,

$$w'_{2n}(x).w'_{2n+1}(x) = w'_1(x)$$

since 2n and 2n+1 are identical except in the first binary place.

Theorem

$$w_i(x).w_j(x) = w_{i\oplus j}(x)$$

Proof

Let the binary expansions of i and j be $d_1d_2 \ldots d_m$ and $f_1f_2 \ldots f_m$, respectively.

Then, from the relationship between the R and W orderings,

$$w_{d_1 d_2 \ldots d_m}(x).w_{f_1 f_2 \ldots f_m}(x) = w'_{d_m\oplus d_{m-1},d_{m-1}\oplus d_{m-2},\ldots,d_2\oplus d_1,d_1}(x).$$

$$w'_{f_m\oplus f_{m-1},f_{m-1}\oplus f_{m-2},\ldots,f_2\oplus f_1,f_1}(x)$$

$$= w'_{d_m\oplus f_m\oplus d_{m-1}\oplus f_{m-1},d_{m-1}\oplus f_{m-1}\oplus d_{m-2}\oplus f_{m-2},\ldots,d_2\oplus f_2\oplus d_1\oplus f_1,d_1\oplus f_1}(x)$$

from the previous theorem

$$= w_{d_1\oplus f_1,d_2\oplus f_2,\ \ldots\ ,d_m\oplus f_m}(x)$$

from the relationship between the orderings again.

As a special case,

$$w_{2n}(x).w_{2n+1}(x) = w_1(x) \quad .$$

# CHAPTER 3

# APPROXIMATION, SPECTRA AND THE FAST WALSH TRANSFORM

## Function approximation

Methods of function approximation which employ algebraic polynomials are widely used, although they are far from being ideally suited to digital computer use because of the large number of multiplications which are involved in the computations. The same criticism also applies to the use of trigonometric functions, and it is only by approximating a given function by means of a linear combination of orthogonal boolean functions, preferably complete in $L^2$, that all multiplications can be avoided in the calculations.

Suppose the interval from 0 to 1 is divided into $N\ (=2^n)$ equal subintervals, where $n$ is any positive integer; and also that $w_{i,j}$ denotes the value of the ith Walsh function, $w_i(x)$, in the jth subinterval.

$f(x)$ is a function to be approximated on the interval from 0 to 1. It is assumed that the integral from 0 to 1 of $f(x)$ exists and is finite.

$f(x)$ is first approximated by means of a step function which takes constant values in each of the $N$ equal subintervals of $[0,1)$ in the following manner:

$$f_k = \frac{f_{max} + f_{min}}{2}$$

where $f_{max}$ and $f_{min}$ are the maximum and minimum values of $f(x)$ in the kth subinterval.

This step function approximation gives the best fit to $f(x)$ if the Walsh functions are subsequently to be used, since the latter are themselves constant in the $N$ subintervals of $[0,1)$.

The step function $f$ can be expressed as a linear combination of the Walsh functions $w_0(x), w_1(x), \ldots, w_{N-1}(x)$:

$$f_k = \sum_{j=0}^{N-1} c_j \cdot w_{k,j} \qquad \text{for} \quad k = 1, 2, \ldots, N \qquad (I)$$

where the $c_j$s are Walsh-Fourier coefficients, which can be computed as the correlation between the step function $f$ and the corresponding Walsh functions:

$$c_i = \frac{1}{N} \sum_{k=1}^{N} f_k \cdot w_{i,k} \qquad \text{for} \quad i = 0, 1, \ldots, N-1 \qquad (II)$$

This summation defines the discrete Walsh transform.

Note that in (I) only the first $N$ Walsh functions are used, since $c_m$, $m > N$, are all equal to zero.

The transform is its own inverse. If (I) is substituted in (II):

$$\frac{1}{N} \sum_{k=1}^{N} \sum_{j=0}^{N-1} c_j \cdot w_{j,k} \cdot w_{i,k}$$

$$= \frac{1}{N} \sum_{j=0}^{N-1} c_j \sum_{k=1}^{N} w_{j,k} \cdot w_{i,k}$$

$$= \frac{1}{N} \sum_{j=0}^{N-1} c_j \cdot d_{i,j}$$

where $d_{i,j} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$

since the Walsh functions are orthogonal

$$= c_i \, .$$

Since the transform is its own inverse, there is no information loss in passing from the step function domain to the Walsh coefficient domain. In this sense, the transform provides a complete description of the function in terms of the coefficients, and the only process of approximation in the true sense of the word is the transformation from the original function $f(x)$ to the step function $f$ .

The transform is readily extended to higher dimensions, although this is not in fact necessary, owing to the mapping of all higher dimensional Walsh functions onto one dimensional Walsh functions, as explained in the following chapter.

By increasing the value of $N$ , the step function - and therefore the coefficients - become a better approximation of $f(x)$. The accuracy of the approximation may be increased as much as desired, since

$$f(x) = \sum_{j=0}^{\infty} c_j . w_j(x) \qquad \text{and} \qquad c_i = \int_0^1 f(x) . w_i(x)\, dx$$

The coefficients $c_j$ tend to zero as $j$ tends to infinity, since

$$\sum_{i=0}^{\infty} c_i^2 = \int_0^1 f^2(x)\, dx$$

This is the analogue of Parseval's relationship for trigonometric series.

Also

$$\sum_{i=0}^{N-1} c_i^2 = \sum_{k=1}^{N} f_k^2$$

The ratio

$$\frac{\sum_{i=0}^{N-1} c_i^2}{\int_0^1 f^2(x)\, dx}$$

may be used as a measure of the accuracy of an approximation.

Note that in (II) on page 26, the calculation of the coefficients by means of the discrete transform, since $w_{i,k}$ always takes the value +1 or -1, the computation involves only the addition or subtraction of the appropriate $f_k$.

Walsh- Fourier spectra

There exist analogues between the Walsh transform and the Fourier transform which suggest the following:

$c_0$ is the total 'intensity' in the function domain.

$c_0$ is the correlation of the step function $f$ with the first Walsh function, $w_0(x)$. Since $w_0(x)$ is unity everywhere, $c_0$ is the sum of the values of the step function.

$f_k^2$ is an intensity spectrum

$\frac{1}{N}\sum_{i=0}^{N-1} c_i^2 = \sum_{k=1}^{N} f_k^2$ is the total energy in the coefficient (function-domain).

$c_i^2$ is an energy spectrum

The first $2^n$ Walsh functions are used in the Walsh transform. In the previous sections of this chapter, the notation $w_i(x)$, rather than $w_i'(x)$, has been used. In so far as the first $2^n$ Walsh functions are the same in both the W and R orderings the distinction is not important. However, in forming the Walsh spectrum, the ordering is not irrelevant.

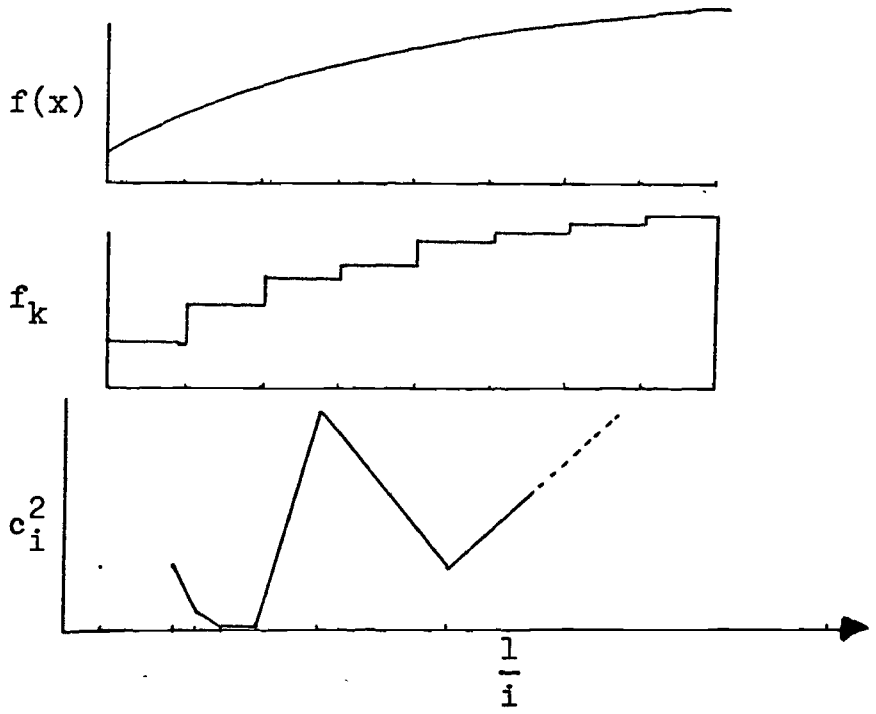
The energy spectrum is obtained by plotting $c_i^2$ against $\frac{1}{i}$. In the W ordering, $w_i(x)$ has exactly $i$ discontinuities in the interval from 0 to 1. The coefficient $c_i$ therefore represents the degree of correlation between the step function $f$ and a standard function with $i$ discontinuities. The value of $i$ may be regarded as the frequency (or sequency (11)) of $w_i(x)$.

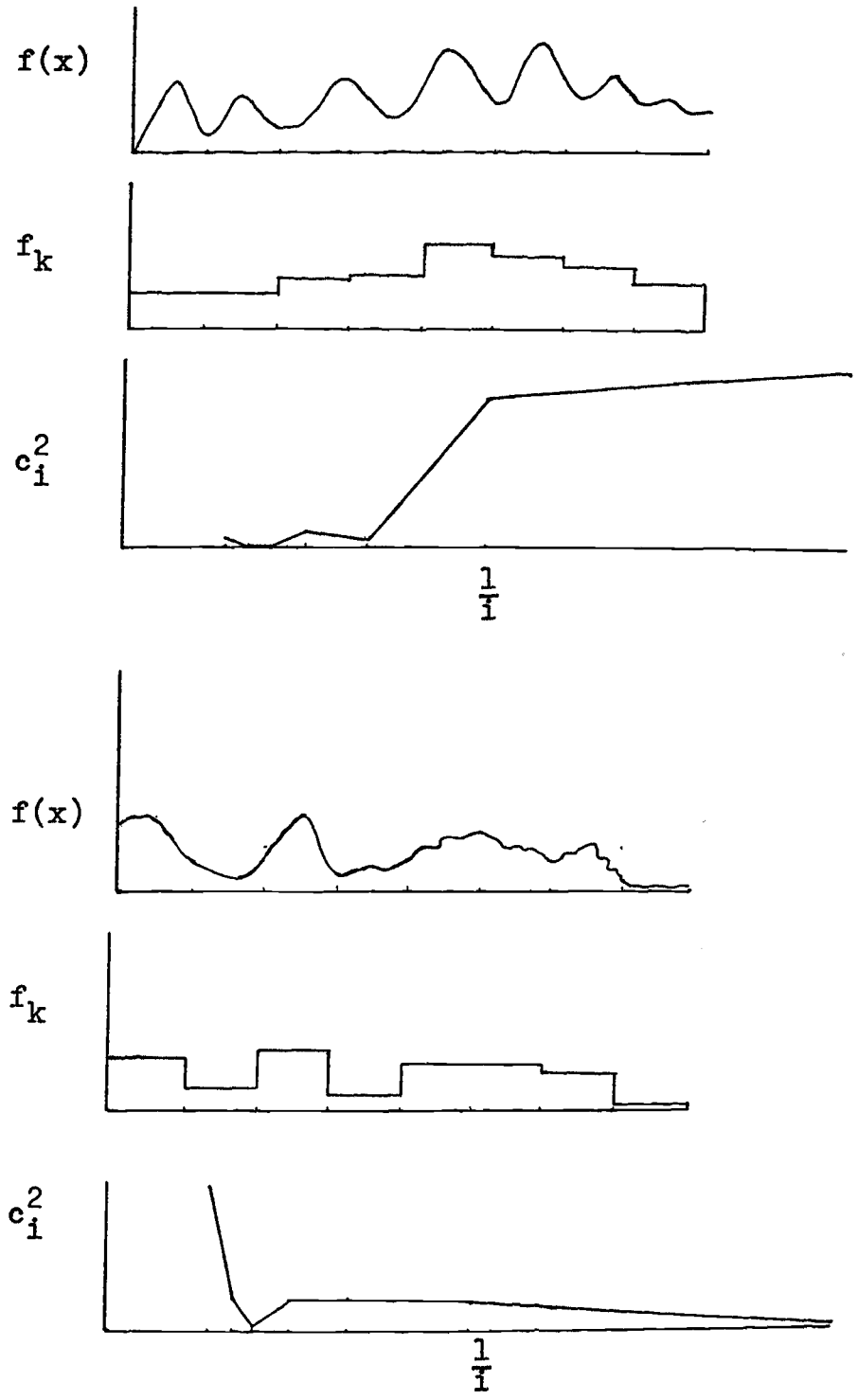
$c_i^2$ is a measure of the similarity between f and $w_i(x)$ or of the dissimilaity between f and $-w_i(x)$ . When plotted against $\frac{1}{i}$ , which may be interpreted as 'wavelength', the result is the Walsh spectrum.

This simple interpretation of the Walsh energy spectrum in terms of wavelength, etc., would not be possible if the R ordering of the functions were used. However, the section of this chapter entitled 'The fast Walsh transform' will show the great computational advantages to be derived from the R ordering. Consequently, the relationship between the two orderings, (which was dealt with in the previous chapter), is of great importance.

Computer programs have been written to carry out the discrete Walsh transform and to plot the spectra resulting from the inputted step functions.

Examples:



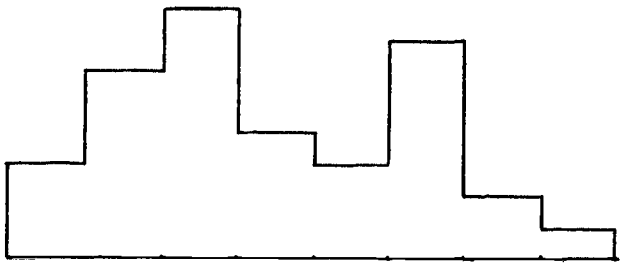



Note how the function with great detail has larger high frequency coefficients.

Clearly, by squaring the coefficients and thus losing information as to whether they were positive or negative, the spectrum does not exactly specify the step function, f , in the way that the coefficients do. However, the spectrum is extremely useful as a visual representation of the result of the transform. Each coefficient contains information about the whole function, and if a single coefficient is altered, when the inverse transform is executed, the change will be seen on a greatly reduced scale throughout the function values, i.e. the perturbation has been spread over the re-constituted function. This is of particular importance in signal transmission.

Example:

$f_k$ :



Walsh transform

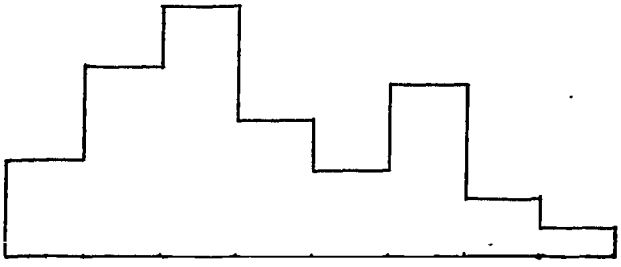
$c_i$ : 34 -2 4 -12 8 4 -10 -2

A coefficient becomes altered

$c_i'$ : 34 -2 4 -12 9 4 -10 -2

Inverse transform

$f_k'$ :



In certain applications, the step function may only take a restricted set of values. In these cases, the alteration of a coefficient may not prevent the step function from being completely re-constituted when the inverse transform is executed.

Example:

The step function may take only integer values.

$f_k$: 7 6 0 3 4 5 -2 1

Walsh transform

$c_i$: 24 -6 20 6 8 2 0 2 x $\frac{1}{8}$

A coefficient becomes altered

$c_i'$: 24 -6 20 $\underline{9}$ 8 2 0 2 x $\frac{1}{8}$

Inverse transform

$f_k'$: 7.4 5.6 -0.4 3.4 4.4 4.6 -2.4 1.4

from which it can be deduced that

$f_k$: 7 6 0 3 4 5 -2 1

The inverse transform referred to above is, of course, simply the Walsh transform applied to coefficients to produce function values rather than vice versa. The actual algorithm employed is identical.

There is often a good reason for wishing to concentrate on a particular part of the energy spectrum. For example, it may be anticipated that differences in signals will be apparent in a certain section of their respective spectra. Also, the high frequency end of the spectrum is often susceptible to the effects of noise. The technique of concentrating on a particular section of the spectrum is referred to as bandwith filtering in signal processing, and is widely used. It is also possible to produce only part of the spectrum, by computing only a selection of the coefficients. When the fast Walsh transform is explained it will be seen that this is no longer possible, but the 'logical' transform retains this feature of flexibility.

The fast Walsh transform

The fast Walsh transform is a computational procedure which has features in common with the fast Fourier transform. It is most easily expounded in terms of the notation of pattern products, which is recapitulated upon here:

$\bar{O}$ is the basic pattern 1 -1

$\bar{I}$ is the basic pattern 1 -1

Example of a pattern product:

$$I.O.I = I.O. \quad 1 \; -1$$
$$= I. \quad 1 \; -1 \; 1 \; -1$$
$$= \quad 1 \; -1 \; 1 \; -1 \; -1 \; 1 \; -1 \; 1$$

Suppose the coefficient corresponding to the Walsh function whose pattern product is, say, $\bar{I}.\bar{O}.\bar{I}$ , is required. The coefficient is

$$\frac{1}{8} ( f_0 - f_1 + f_2 - f_3 - f_4 + f_5 - f_6 + f_7 )$$

where the $f_k$ are the step function values, and the plusses and minuses correspond to the values, the +1s and -1s , of the function $\bar{I}.\bar{O}.\bar{I}$ .

Now

$$f_0 - f_1 + f_2 - f_3 - f_4 + f_5 - f_6 + f_7$$

$$= \quad ( f_0-f_1 ) + ( f_2-f_3 ) - ( f_4-f_5 ) + ( f_6-f_7)$$

which is a difference of sums of differences of the step function values, $f_0, f_1, \ldots, f_7$ .

This is denoted as DSD - a D denoting a difference and an S a sum. Note that there is a direct relationship between the Ds and the $\bar{I}$s of the pattern product and between the Ss and the $\bar{O}$s .

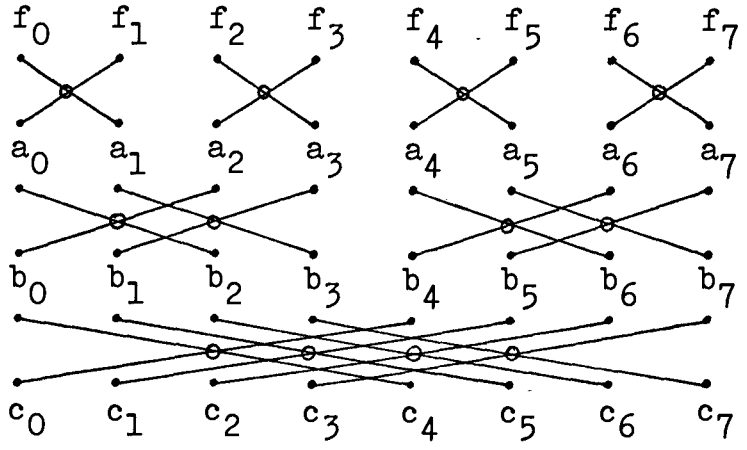
The Walsh functions are all possible pattern products, and so the coefficients can be computed as all the possible sums of differences and differences of sums of the step function values. Hence, the $2^n$ coefficients can be computed as below. Note how full use is made of partial results.

Example: for $n = 3$

| step function values | Calculation of coefficients | | |
|---|---|---|---|
| | stage 1 | stage 2 | stage 3 |
| $f_0$ | $a_0=f_0+f_1$ | $b_0=a_0+a_2$ | $c_0=b_0+b_4$ |
| $f_1$ | $a_1=f_0-f_1$ | $b_1=a_1+a_3$ | $c_1=b_1+b_5$ |
| $f_2$ | $a_2=f_2+f_3$ | $b_2=a_0-a_2$ | $c_2=b_2+b_6$ |
| $f_3$ | $a_3=f_2-f_3$ | $b_3=a_1-a_3$ | $c_3=b_3+b_7$ |
| $f_4$ | $a_4=f_4+f_5$ | $b_4=a_4+a_6$ | $c_4=b_0-b_4$ |
| $f_5$ | $a_5=f_4-f_5$ | $b_5=a_5+a_7$ | $c_5=b_1-b_5$ |
| $f_6$ | $a_6=f_6+f_7$ | $b_6=a_4-a_6$ | $c_6=b_2-b_6$ |
| $f_7$ | $a_7=f_6-f_7$ | $b_7=a_5-a_7$ | $c_7=b_3-b_7$ |

Consider the fourth row: first $a_3$ is formed, the difference between $f_2$ and $f_3$ ; then the difference $a_1 - a_3$ which is a difference of differences ( since $a_1$ is the difference $f_0 - f_1$ ); and finally the sum of $b_3$ and $b_7$ which is therefore a sum of differences of differences since both $b_3$ and $b_7$ are differences.

The computation can be represented diagrammatically:



Working upwards from the encircled intersections, the two values which are to be summed and differenced are found. Their results are the end points of the lines below the circle.

To compute all $2^n$ coefficients only $n.2^n$ additions and subtractions are required, compared with a total of $2^n.2^n$ for the discrete Walsh transform. Each of the $c_i$ must be scaled by a factor of $2^n$ ( in this case 8 ), and even this is a shift in a binary computer.

Example:

| | | | |
|---|---|---|---|
| $f_0 = 0.2$ | $a_0 = 0.6$ | $b_0 = 2.0$ | $c_0 = 4.0$ |
| $f_1 = 0.4$ | $a_1 = -0.2$ | $b_1 = -0.4$ | $c_1 = 0.8$ |
| $f_2 = 0.6$ | $a_2 = 1.4$ | $b_2 = -0.8$ | $c_2 = -1.2$ |
| $f_3 = 0.8$ | $a_3 = -0.2$ | $b_3 = 0.0$ | $c_3 = 0.0$ |
| $f_4 = 0.7$ | $a_4 = 0.8$ | $b_4 = 2.0$ | $c_4 = 0.0$ |
| $f_5 = 0.1$ | $a_5 = 0.6$ | $b_5 = 1.2$ | $c_5 = -1.6$ |
| $f_6 = 0.9$ | $a_6 = 1.2$ | $b_6 = -0.4$ | $c_6 = -0.4$ |
| $f_7 = 0.3$ | $a_7 = 0.6$ | $b_7 = 0.0$ | $c_7 = 0.0$ |

and hence the Walsh coefficients are

4.0 0 -0.4 -1.2 0 0 -1.6 0.8

The fast Walsh transform can be modified so that it becomes equivalent to the repeated execution of a much simpler algorithm, in the same way that the fast Fourier transform can. The calculations at each stage of the transform can be carried out in parallel, since the

production of, say, $b_0$ and $b_2$ from $a_0$ and $a_2$, is quite independent of the remaining work at that stage. However, only when the transform is modified so that each stage of the computation is identical in format can the construction of a special purpose machine to carry out the transform in parallel be considered a realistic proposition from an economic point of view, (22, 23).

The simplified algorithm is defined below, and each execution of it is equivalent to one stage of the fast Walsh transform as defined above. It must therefore be carried out n times to produce $2^n$ coefficients. (It is perhaps worth mentioning here that, since the transform is its own inverse, the coefficients that would be produced after 2n stages, or after 2n executions of the modified transform, would be the original step function values, apart from the scaling factor.)

A single stage of the modified algorithm is defined by the following equations:

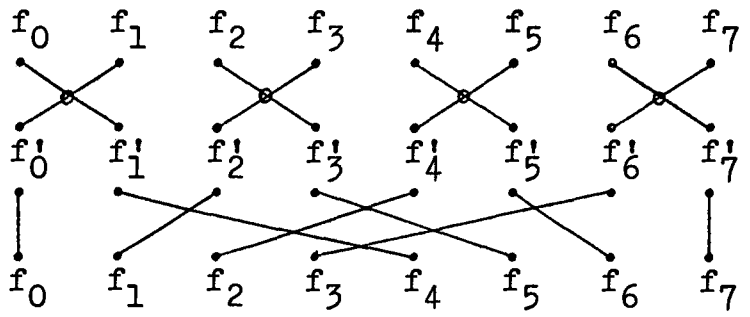
$$
\begin{aligned}
f'_0 &= f_0 + f_1 \\
f'_1 &= f_0 - f_1 \\
f'_2 &= f_2 + f_3 \\
f'_3 &= f_2 - f_3 \\
f'_4 &= f_4 + f_5 \\
f'_5 &= f_4 - f_5 \\
f'_6 &= f_6 + f_7 \\
f'_7 &= f_6 - f_7
\end{aligned}
\qquad \text{followed by} \qquad
\begin{aligned}
f_0 &= f'_0 \\
f_1 &= f'_2 \\
f_2 &= f'_4 \\
f_3 &= f'_6 \\
f_4 &= f'_1 \\
f_5 &= f'_3 \\
f_6 &= f'_5 \\
f_7 &= f'_7
\end{aligned}
$$

or equivalently:

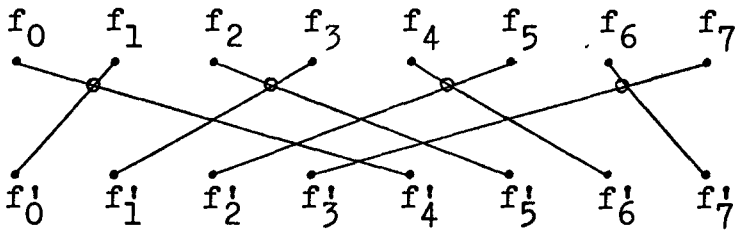
$$
\begin{aligned}
f'_0 &= f_0 + f_1 \\
f'_1 &= f_2 + f_3 \\
f'_2 &= f_4 + f_5 \\
f'_3 &= f_6 + f_7 \\
f'_4 &= f_0 - f_1 \\
f'_5 &= f_2 - f_3 \\
f'_6 &= f_4 - f_5 \\
f'_7 &= f_6 - f_7
\end{aligned}
\qquad \text{followed by} \qquad f_i = f'_i \quad \text{for all } i
$$

Diagrammatically, these two versions may be represented as follows:

(i)




(ii)




The fast Walsh transform, and its modified form, can be extended to higher dimensions, but this is not necessary, as has already been mentioned.

The coefficients produced by the fast Walsh transform are in the R ordering, since the pattern product representation of the functions was used. So that meaningful spectra can be obtained, it is usually required that the coefficients be in the W ordering. The permutation, based upon the relationship between the two orderings which was examined in the previous chapter, is effected by means of an algorithm (below). The algorithms for the fast Walsh transform and its modified version are also given.

Algorithms

1. Permutation of R and W orderings. Given n and j, the algorithm finds k such that $w_k(x) = w'_j(x)$, where $j < 2^n$, $k < 2^n$.

```
k = and(bits(j),1)
cycle i = 1, 1, n-1
  if and(bits(and(j,shift(n,1-i))),1) = 0 then goto 1
  k = k + shift(1,i)
1:repeat
  print(k)
  end
```

shift(m,n) has the effect of moving the binary representation of m to the left by n places.

and(m,n) has the effect of producing a number whose binary representation has unity in those places in which both m and n had unity.

bits(m) has the value of the number of bits in the binary representation of m which are unity.

2. The fast Walsh transform. Given data in an array W of dimensions $1 \times 2^n$, the algorithm produces the Walsh coefficients (unscaled) in the same array, in the R ordering.

```
   k = 0
1:m = shift(1,k)
   cycle i = 0, 2m, 2^n-2m
     cycle j = 0, 1, m-1
       p = W(i+j)
       q = W(i+j+1)
       W(i+j) = p + q
       W(i+j+1) = p - q
     repeat
   repeat
   k = k + 1
   goto 1 unless k = n
   end
```

3. The modified fast Walsh transform. Exactly as above.

```
   m = 2^n
   cycle i = 0, 1, n-1
1:cycle j = 0, 2, (m-1)/2
       k = shift(j,-1)
       g(k) = W(j) + W(j+1)
       g(k+m/2) = W(j) - W(j+1)
     repeat
     swaparrays (g,W)        comment: puts the contents of
                                      g into W and vice versa.
   repeat
   end
```

# CHAPTER 4

# THE LOGICAL WALSH TRANSFORM

## The inadequacies of the fast Walsh transform

The Walsh functions are essentially binary valued, taking only the values +1 and -1. Since the fast Walsh transform will most frequently be implemented on a binary computer, the data will also be in binary form. In these circumstances, there are two respects in which it appears that useful modifications could be made to the transform.

Firstly, the transform produces a set of coefficients which contain no more information than the data, but some of which may contain more bits than a single datum. Example:

| | | | | |
|---|---|---|---|---|
| Data: | 11 | 10 | 00 | 11 |
| Coefficients: | 1000 | -10 | 10 | 100 |

If each datum occupies one computer word, the coefficients may require more space, unless the extra bits can be eliminated. This creates difficult storage problems.

Secondly, both the Walsh functions and the data are of a binary nature, whilst the computation of the fast Walsh transform is arithmetic, rather than boolean, which, intuitively, would be more appropriate.

It will be shown how the extra bits in the Walsh coefficients can be avoided, and also how the transform can be

altered to a non-arithmetic form which produces the coefficients without the extra bits directly.

Binary coefficients

The data is assumed to consist of $N$ $(=2^n)$ bits, from which $N$ coefficients are to be produced by means of the fast Walsh transform. Clearly, if the coefficients are to contain no more bits than the data, then each must take either of only two possible values. The obvious choice for the set of values which the coefficients may take is $\{0, 1\}$ . In this case, the transform from data to coefficients constitutes a mapping of the first $2^N$ binary numbers into, and in fact onto, themselves. (If the data consists of $N$ bits, then there are $2^N$ possible sets of data).

It will be shown that all binary data can be reduced to the form where each datum consists of a single bit.

There is one further restriction which must be placed on the format of the data at this stage. That is, that the first datum must be unity, rather than zero. This is because the Walsh functions take the values $+1$ and $-1$ with equal frequency, except for the first Walsh function, $w_0(x)$ , which is unity everywhere. The imbalance is compensated for here by placing the above condition on the first datum.

The logical transform

The Walsh coefficients are computed in the normal manner by means of the fast Walsh transform. They are then reduced to binary coefficients, 1 or 0 , depending upon whether the original value is positive or non-positive, respectively.

Example:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Data: | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 |
| Walsh coeffs: | 5 | -1 | -1 | 1 | 1 | 3 | -1 | 1 |
| Binary coeffs: | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 |

Note that the first Walsh coefficient will always be positive, since it is simply the sum of the data bits (which cannot all be zero, because the first datum must be unity), and consequently the first binary coefficient will always be unity.

It is conjectured that the logical transform above is its own inverse. A proof of this, and equivalent conjectures in related fields, have been attempted by many workers without success.

There are two further, related, conjectures. Firstly, one which is stronger than the conjecture above:

If the logical transform is applied to real, rather than binary, data; and the logical transform is applied again to the resulting binary coefficients, then the final binary coefficients will be equal to those obtained by applying an appropriate threshold function to the original real data.

Example:

| | | | | |
|---|---|---|---|---|
| Data: | 5 | 2 | -1 | 0 |
| Binary coeffs: | 1 | 1 | 1 | 1 |
| Binary coeffs: (using the above as data) | 1 | 0 | 0 | 0 |

If any value of $T$ had been chosen such that $2 < T < 5$, and the data reduced according as it were greater or less than $T$, the final binary coefficients would have been obtained directly from the data.

Binary data is a special case from which the first conjecture - that the logical transform is its own inverse - can be inferred.

Secondly, a conjecture about the logical transform which corresponds to the following statement about the discrete Walsh transform:

$$\sum_{i=0}^{N-1} (c_i - c_i')^2 = \sum_{i=0}^{N-1} (f_i - f_i')^2$$

where $c_i$ and $c_i'$ are typical coefficients produced by the Walsh transform from the step functions $f$ and $f'$ respectively.

For the logical transform, if $x_1, x_2, \ldots, x_N$ and $y_1, y_2, \ldots, y_N$ are two sets of binary data, and $c_1, c_2, \ldots, c_N$ and $d_1, d_2, \ldots, d_N$ are the corresponding sets of binary coefficients, then

$$\sum_{i=1}^{N} (c_i - d_i)^2 = \sum_{i=1}^{N} (x_i - y_i)^2$$

or bits(nq(C,D)) = bits(nq(X,Y))

where nq is bit-by-bit non-equivalence

and bits is the number of 1s in the argument.

Example:

The inverse of the logical transform.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| data: | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |
| Walsh coeffs: | 4 | 0 | 2 | 2 | 0 | 0 | 2 | -2 |
| binary coeffs: | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 |

using the above as data

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Walsh coeffs: | 4 | 2 | -2 | 0 | 2 | 0 | 0 | 2 |
| binary coeffs: | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |

The reason that a Walsh coefficient is reduced to a binary coefficient of zero if it is itself zero is again that there is an imbalance between positive and negative values of the Walsh functions.

Removal of restrictions on format of data

If the first of the N data is zero, rather than unity, the procedure is as follows:

the data is complemented, then the Walsh coefficients computed, then the binary coefficients, and finally the binary coefficients are complemented.

Example:

| | | | | |
|---|---|---|---|---|
| data: | 0 | 1 | 1 | 1 |
| complemented: | 1 | 0 | 0 | 0 |
| Walsh coeffs: | 1 | 1 | 1 | 1 |
| binary coeffs: | 1 | 1 | 1 | 1 |
| complemented: | 0 | 0 | 0 | 0 |

When the orignal data is complemented, the first datum becomes unity, producing a first binary coefficient of unity. After the final complementation, the first binary coefficient becomes zero. Hence, the first binary coefficient is always equal to the first datum. Also, of the $2^N$ possible sets of data, the first $2^{N-1}$ map into themselves, and the second $2^{N-1}$ into themselves.

If each of the N data contains m bits rather than just one, the procedure is as follows:

the logical transform is applied to m sets of data, each of which contains N bits - the jth set of data consisting of the jth bit from each of the N m-bit data. In each case, the resulting binary coefficients replace, positionally, the data from which they were derived.

Example:

| | | | | | |
|---|---|---|---|---|---|
| data : | 101 | 101 | 100 | 010 | |
| | 1 | 1 | 1 | 0 | 1st data set |
| | 0 | 0 | 0 | 1 | 2nd data set |
| | 1 | 1 | 0 | 0 | 3rd data set |
| binary coeffs: | 1 | 1 | 1 | 0 | 1st coeff set |
| | 0 | 1 | 1 | 1 | 2nd coeff set |
| | 1 | 0 | 1 | 0 | 3rd coeff set |
| | 101 | 110 | 111 | 010 | |

Note that the binary coefficients produced from N m-bit data are different from those produced from the application of the logical transform to N.m single bit data.

Example:

| | | | | |
|---|---|---|---|---|
| data: | 10 | 11 | 00 | 10 |
| | 1 | 1 | 0 | 1 |
| | 0 | 1 | 0 | 0 |
| binary coeffs: | 1 | 1 | 1 | 0 |
| | 0 | 1 | 0 | 0 |
| | 10 | 11 | 10 | 00 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| data: | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 |
| binary coeffs: | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 |

Some applications of the Walsh functions, particularly in the field of pattern recognition, involve the use of two- or even higher- dimensional Walsh functions and transforms. In general the n dimensional transform can be reduced quite simply to the one-dimensional case.

Suppose the array below represents the values of a two-dimensional binary function:

| | | | |
|---|---|---|---|
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 |

Then the two-dimensional transform applied to this array will produce the same Walsh, and therefore binary, coefficients, as the one-dimensional transform applied to

1 0 1 1 1 1 0 0 0 1 0 1 1 0 0 1

i.e. the set of data produced if the rows of the array are written out one after the other.

The same coefficients would also be produced if the columns of the array were written out one after the other, since to each two-dimensional Walsh function, and therefore coefficient, with indices i, j there corresponds another with indices j, i .

The mapping from the indices of the resultant two-dimensional binary coefficients to the index of the appropriate one-dimensional coefficient depends, of course, upon whether the array is read by rows or columns. This problem is considered in practice in Chapter 6.

The two-dimensional Walsh function

$$\begin{aligned} w^{!}_{i,j}(x,y) &= w^{!}_{i}(x).w^{!}_{j}(y) \\ &= w^{!}_{d_1 d_2 \,..\, d_m}(x).w^{!}_{f_1 f_2 \,..\, f_m}(y) \\ &= w^{!}_{d_1 d_2 \,..\, d_m f_1 f_2 \,..\, f_m}(x) \\ &= w^{!}_{f_1 f_2 \,..\, f_m d_1 d_2 \,..\, d_m}(y) \end{aligned}$$

where $d_1 d_2 \,..\, d_m$ and $f_1 f_2 \,..\, f_m$ are the binary expansions of $i$ and $j$ respectively.

Example:

$$w'_{3,2}(x,y) = w'_{11}(x).w'_{10}(y)$$
$$= w'_{1110}(x) = w'_{14}(x)$$
$$= w'_{1011}(y) = w'_{11}(y)$$

The reason for the mapping of two- and higher-dimensional Walsh functions onto one-dimensional Walsh functions ( and therefore of coefficients onto coefficients) is that an n-dimensional Walsh function is the product of n one-dimensional Walsh functions, which can be multiplied together under a different interpretation to form another one-dimensional function.

Example:

Walsh functions:

| | | | | |
|---|---|---|---|---|
| 1 | 1 | -1 | -1 | $w'_2(x)$ |
| 1 | -1 | -1 | 1 | $w'_3(x)$ |

Multiplied together to give two-dimensional functions, $w'_{2,3}(x,y)$ and $w'_{3,2}(x,y)$ :

| | | | |
|---|---|---|---|
| 1 | -1 | -1 | 1 |
| 1 | -1 | -1 | 1 |
| -1 | 1 | 1 | -1 |
| -1 | 1 | 1 | -1 |

and

| | | | |
|---|---|---|---|
| 1 | 1 | -1 | -1 |
| -1 | -1 | 1 | 1 |
| -1 | -1 | 1 | 1 |
| 1 | 1 | -1 | -1 |

Multiplied together to give one-dimensional functions, $w'_{14}(x)$ and $w'_{11}(x)$ :

1 1 -1 -1 -1 -1 1 1 -1 -1 1 1 1 1 -1 -1

1 -1 -1 1 1 -1 -1 1 -1 1 1 -1 -1 1 1 -1

The difference between the two types of product is also shown by the following:

$$w'_{d_1 d_2 .. d_m}(x) . w'_{f_1 f_2 .. f_m}(x) = w'_{d_1 \oplus f_1, d_2 \oplus f_2, .., d_m \oplus f_m}(x)$$

$$w'_{d_1 d_2 .. d_m}(x) . w'_{f_1 f_2 .. f_m}(y) = w'_{d_1 d_2 .. d_m f_1 f_2 .. f_m}(x)$$

The method of storing an array in a computer shows an analagous similarity between the n-dimensional case and the one-dimensional case.

A logical algorithm

The logical Walsh transform (based upon the binary valued Walsh functions) operates upon binary data to produce binary coefficients. However, the computation is arithmetic, involving the fundamental operations of the fast Walsh transform - addition and subtraction of two data - before the positive - non-positive criterion is applied. An attempt to find two binary operators, which would each produce a binary result when applied to a pair of bits, and which would replace addition and subtraction in the fast Walsh transform, has proved to be impossible, if the desirable properties of the transform are to be preserved.

Corresponding to the criterion of whether a Walsh coefficient is positive or not, there exists a boolean threshold function, (27). The binary coefficients can be produced by means of a set of such functions, so that, not only do the final coefficients contain no extra bits, but no extra bits are generated in the process of the transform.

The boolean threshold functions simply evaluate, of those data bits which are unity, whether a majority are to be multiplied by +1 in the correlation of the data with the corresponding Walsh function, or by -1.

Suppose there are four data bits, $x_1$, $x_2$, $x_3$, $x_4$. Then the binary coefficient corresponding to the Walsh function which takes the value +1 in the first half of the interval and -1 in the second is given by

$$\bar{x}_2.\bar{x}_3.\bar{x}_4 + x_2.x_3.\bar{x}_4 + x_2.\bar{x}_3.x_4 + x_2.\bar{x}_3.\bar{x}_4$$

where '_' , '.' , and '+' denote boolean negation, conjunction and disjunction, respectively. It is assumed that $x_1$ is unity.

If the negated components of the four disjuncts are replaced by 0s and the remainder by 1s , then the binary numbers from 0 to 3 , or their complements, are obtained from the four disjuncts. This is true in general; when there are N data bits, and for all the functions corresponding to the N binary coefficients, the disjuncts will represent the binary numbers from 0 to N-1 or their complements.

Suupose the data is 1 1 0 1 . Then it can be determined whether each binary coefficient is 0 or 1 by discovering whether 1 0 1 ($x_2.\bar{x}_3.x_4$) appears in each of the four expressions corresponding to the coefficients, either as $x_2.\bar{x}_3.x_4$ or as $\bar{x}_2.x_3.\bar{x}_4$ (complemented form).

The Walsh functions can be represented by 1s and 0s , simply by replacing the naturally occuring -1s by 0s . The kth value of the jth Walsh function is denoted by $w_{k,j}$

Consider the following table:

| | $w_1$ | $w_2$ | $w_3$ |
|---|---|---|---|
| Walsh functions | 1 1 -1 -1 | 1 -1 -1 1 | 1 -1 1 -1 |
| Coeff functions | $\bar{x}_2.\bar{x}_3.\bar{x}_4$ | $x_2.\bar{x}_3.x_4$ | $x_2.x_3.\bar{x}_4$ |
| | $+ x_2.x_3.\bar{x}_4$ | $+ \bar{x}_2.x_3.x_4$ | $+ \bar{x}_2.\bar{x}_3.\bar{x}_4$ |
| | $+ x_2.\bar{x}_3.x_4$ | $+ \bar{x}_2.\bar{x}_3.\bar{x}_4$ | $+ \bar{x}_2.x_3.x_4$ |
| | $+ x_2.\bar{x}_3.\bar{x}_4$ | $+ \bar{x}_2.\bar{x}_3.x_4$ | $+ \bar{x}_2.x_3.\bar{x}_4$ |

Relpacing -1s by 0s in the Walsh functions, and components of disjuncts by 1s and 0s depending upon whether they are negated or not, the table above becomes:

| | $w_1$ | $w_2$ | $w_3$ |
|---|---|---|---|
| Walsh functions | 1100 | 1001 | 1010 |
| Coeff functions | 000 | 101 | 110 |
| | 110 | 011 | 000 |
| | 101 | 000 | 011 |
| | 100 | 001 | 010 |

The kth binary coefficient $c_k$ is 1 if the data bits $x_2.x_3.x_4$ appear in the kth column of the above table, and 0 otherwise.

The kth column of the above table is determined by the first column according to the non-equivalence of the bits of $w_1$ and $w_k$. Therefore $c_k$ is 1 if, and only if,

$$nq(x_2x_3x_4, nq(w_{1,2}w_{1,3}w_{1,4}, w_{k,2}w_{k,3}w_{k,4})) \quad (I)$$

appears in the first column

where nq denotes bit-by-bit non-equivalence.

Example:

011 appears in column 3

Therefore

nq(011,nq(100,010)) will appear in column 1

This is in fact the case, since the above expression is equal to 101.

From Chapter 2, it is known that

$$w'_{2n}(x).w'_{2n+1}(x) = w'_1(x)$$

and $$w_{2n}(x).w_{2n+1}(x) = w_1(x)$$

So, whichever ordering of the Walsh functions, R or W, is used

$$eq(w_{1,2}w_{1,3}w_{1,4}, w_{2k,2}w_{2k,3}w_{2k,4}) =$$

$$w_{2k+1,2}w_{2k+1,3}w_{2k+1,4}$$

and $$eq(w_{1,2}w_{1,3}w_{1,4}, w_{2k+1,2}w_{2k+1,3}w_{2k+1,4}) =$$

$$w_{2k,2}w_{2k,3}w_{2k,4}$$

where eq is bit-by-bit equivalence, i.e.

$$eq(0,1) = eq(1,0) = 0 \quad \text{and} \quad eq(0,0) = eq(1,1) = 1$$

From (I) $c_{2k+1}$ is 1 if and only if

$eq(x_2x_3x_4 , w_{2k,2}w_{2k,3}w_{2k,4})$ is in column 1

and $c_{2k}$ is 1 if and only if

$eq(x_2x_3x_4 , w_{2k+1,2}w_{2k+1,3}w_{2k+1,4})$ is in column 1

Example:

Data 1 1 0 0

$c_1$ : eq(100,111) = 100 $c_1 = 1$

$c_2$ : eq(100,010) = 001 $c_2 = 0$

$c_3$ ; eq(100,001) = 010 $c_3 = 0$

The complete set of binary coefficients is 1 1 0 0 , since $c_0$ is always unity.

There are several simple methods by which it can be determined whether or not a particular binary number appears in the first column or not.

For example, storing only four bits, 1 0 0 0 , indicates that

| | |
|---|---|
| 0 0 0 | does appear |
| 0 0 1 | does not appear |
| 0 1 0 | does not appear |
| 0 1 1 | does not appear |

and therefore that

| | |
|---|---|
| 1 1 1 | does not appear |
| 1 1 0 | does appear |
| 1 0 1 | does appear |
| 1 0 0 | does appear |

Computation

The data is held in a computer word, D . The Walsh functions are stored, in their binary form, in an array, A .

A(2k) is a computer word containing $w_{2k+1}$

A(2k+1) is a computer word containing $w_{2k}$ .

The first datum is assumed to be unity, and is omitted. The first bit of each Walsh function is also omitted as it is always unity, so that the array A consists, in general, of $2^n$ words, each of length $2^n-1$ bits.

The array F is a word, each bit of which is an element of F .

$$F(k) = \begin{cases} 1 & \text{if } k \text{ appears as a disjunct of the binary coefficient function corresponding to } w_1 \\ 0 & \text{otherwise} \end{cases}$$

and $F(\bar{k}) = \overline{F(k)}$

for $k = 0, 1, \ldots, 2^n-1$

so that F has in all $2^{n+1}-1$ elements.

Each element of the array A is taken with D , and eq is applied. Then the coefficients are formed in an array C by assigning

$$C(k) = F(A(k))$$

Example:

$$D = 1\ 0\ 1\ 0$$

$$F = 1\ 0\ 0\ 0\ 1\ 1\ 1\ 0$$

$$A = \begin{matrix} 1\ 1\ 1 \\ 0\ 1\ 0 \\ 0\ 0\ 1 \end{matrix}$$

$$A = eq(D,A) = \begin{matrix} 0\ 1\ 0 \\ 1\ 1\ 1 \\ 1\ 0\ 0 \end{matrix}$$

$$F(A) = \begin{matrix} 0 \\ 0 \\ 1 \end{matrix}$$

binary coefficients : 1 0 0 1

A less efficient algorithm for the logical transform was given in (26) .

The fast Walsh transform involves $2^n.n$ additions and subtractions, where there are $2^n$ data values. This logical transform requires only $2^n$ short logical operations in its basic form. However, unlike the fast Walsh transform in which the values of the Walsh functions are implicit, it is necessary to store the Walsh functions in the array, A . This is most easily done by using the result of Chapter 2, which showed how the Walsh functions could be obtained as the scalar products of the binary numbers.

The logical transform retains the advantage of the modified fast Walsh transform that it can be carried out largely in parallel, since each coefficient is computed independently from the remainder. Also, there is only one stage to the logical transform, compared with $n$ for the fast Walsh transform, which greatly simplifies the engineering problem of constructing a special purpose machine to carry out the transform.

The fast Walsh transform produces all the coefficients simultaneously at the final stage of the computation. This can be a disadvantage if only a few coefficients are required. The logical transform does not suffer from this disadvantage.

The logical energy spectrum

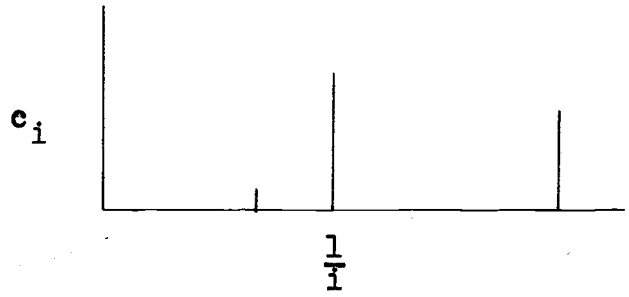
The Walsh energy spectrum plots $c_i^2$ against $\frac{1}{i}$. That is, the positive-negative aspect of the coefficients is neglected, and only their magnitude is taken account of. In the logical energy spectrum, the only information which can be included is the positive-negative information, since that is all that the coefficients themselves contain. (Also, of course, $c_i$ does not tend to zero as $i$ tends to infinity, when $c_i$ is a binary coefficient.) Consequently, the interpretation of the logical spectrum is quite different. This is of no great importance, since the energy spectrum is little more than a convenient means of presenting the result of the transform visually. Users rapidly adjust to a

different visual presentation. In fact, the logical transform spectrum has the advantage over the fast Walsh transform spectrum that no information loss occurs in composing the spectrum from the binary coefficients.

When the data for the logical transform are each of m bits, rather than just 1 , then the logical spectrum will also be more varied.

Example:

| | | | | |
|---|---|---|---|---|
| Data | 101 | 101 | 100 | 010 |
| Binary coeffs | 101 | 110 | 111 | 010 |




logical transform spectrum

The set of binary coefficients can be extremely useful in the comparison of two sets of data in a quantitative manner.

Example:

| | data | | binary coefficients |
|---|---|---|---|
| $f_1$ | 1 0 0 1 1 0 1 1 | $c_1$ | 1 0 1 0 1 1 0 1 |
| $f_2$ | 1 0 0 1 0 1 0 1 | $c_2$ | 1 0 0 0 1 1 1 0 |
| $nq(f_1,f_2)$ | 0 0 0 0 0 1 1 0 | $nq(c_1,c_2)$ | 1 0 0 1 1 1 1 1 |

If $f_1$ and $f_2$ are treated as binary numbers, and the dissimilarity between them is represented by $nq(f_1,f_2)$, then it would be meaningless to interpret the latter as a binary number whose magnitude is a measure of the difference between the two data sets, since undue emphasis is put upon, say, the first bit of the number, owing solely to its position. However, there is every justification for interpreting $nq(c_1,c_2)$ as a binary number whose magnitude is a measure of the difference between the two data sets, since the first bit of that number represents the difference in correlation between the two data sets and a standard function with one discontinuity in the interval in the W ordering, the second bit represents the difference in correlation of the two data sets with a standard function which has two discontinuities in the interval, etc. On this basis, a case can be made out for plotting the logical spectrum with $c_i$, the binary coefficients, against $\frac{1}{2^i}$.

# CHAPTER 5

## HAAR FUNCTIONS

Walsh developed his functions from the Haar functions and showed how a Walsh function could be expressed as a linear combination of a finite number of Haar functions, and vice versa. The Haar functions are a complete orthogonal set of functions, but unlike the Walsh functions they take more than two values. Even omitting normalising factors, they take three values - 0, +1, and -1. In fact, the Walsh functions are the only complete orthogonal set of binary-valued functions which can be extended indefinitely in size. However, the Haar functions do have some value, both in terms of computational speed and suitability to particular problems.

### Definition

The Haar functions are denoted by $h_0(x)$, $h_1(x)$, . . .

$$h_0(x) = 1 \qquad 0 \leqslant x < 1$$

$$h_1(x) = \begin{cases} 1 & 0 \leqslant x < \frac{1}{2} \\ -1 & \frac{1}{2} \leqslant x < 1 \end{cases}$$

$$h_2^1(x) = \begin{cases} \sqrt{2} & \\ -\sqrt{2} & \\ 0 & \\ 0 & \end{cases} \qquad h_2^2(x) = \begin{cases} 0 & 0 \leqslant x < \frac{1}{4} \\ 0 & \frac{1}{4} \leqslant x < \frac{1}{2} \\ \sqrt{2} & \frac{1}{2} \leqslant x < \frac{3}{4} \\ -\sqrt{2} & \frac{3}{4} \leqslant x < 1 \end{cases}$$

and in general

$$h_n^k(x) = \begin{cases} \sqrt{2^{n-1}} & \frac{k-1}{2^{n-1}} \leqslant x < \frac{2k-1}{2^n} \\ -\sqrt{2^{n-1}} & \frac{2k-1}{2^n} \leqslant x < \frac{k}{2^{n-1}} \\ 0 & 0 \leqslant x < \frac{k-1}{2^{n-1}} \quad \text{or} \quad \frac{k}{2^{n-1}} \leqslant x < 1 \end{cases}$$

where $k = 1, 2, \ldots, 2^{n-1}$

$n = 1, 2, \ldots$

With normalising factors omitted, the first eight Haar functions can be represented by an array of 0, +1 and -1 values:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | -1 | -1 | -1 | -1 |
| 1 | 1 | -1 | -1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | 1 | -1 | -1 |
| 1 | -1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | -1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | -1 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | -1 |

It is assumed in the above ordering of the first eight Haar functions that the two indices of the definition have been reduced to a single index as was done for the Walsh functions.

Two-dimensional Haar functions

Unlike the Walsh functions, two- and higher-dimensional Haar functions do not map onto the one-dimensional Haar functions. Two-dimensional Haar functions, for example, can be formed in two quite different ways:

(i)

$$h_i(x).h_j(y) = h_{i,j}(x,y)$$

Example:

$$h_{1,2}(x,y) = \begin{matrix} 1 & 1 & -1 & -1 \\ -1 & -1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{matrix}$$

(ii) The two-dimensional functions can be formed by writing out the one-dimensional functions by rows or columns. This is the method which, for Walsh functions, produces the same result as method (i) above.

Example:

$$\begin{aligned} h_0 &= 1 \quad 1 \quad 1 \quad 1 \\ h_1 &= 1 \quad 1 \quad -1 \quad -1 \\ h_2 &= 1 \quad -1 \quad 0 \quad 0 \\ h_3 &= 0 \quad 0 \quad 1 \quad -1 \end{aligned}$$

from which,

$$h_{0,0} = \begin{matrix} 1 & 1 \\ 1 & 1 \end{matrix} \qquad h_{0,1} = \begin{matrix} 1 & 1 \\ -1 & -1 \end{matrix}$$

$$h_{1,0} = \begin{matrix} 1 & -1 \\ 0 & 0 \end{matrix} \qquad h_{1,1} = \begin{matrix} 0 & 0 \\ 1 & -1 \end{matrix}$$

In both cases (i) and (ii) the two-dimensional Haar functions are orthogonal. For higher dimensions, there is a greater variety of ways in which the functions can be formed, and a choice can be made of the most appropriate for a particular application.

Example:

(a) $h_{1,0,1}(x,y,z) = h_1(x).h_0(y).h_1(z)$

$$= \begin{matrix} 1 & -1 \\ -1 & -1 \end{matrix} \quad \begin{matrix} -1 & 1 \\ -1 & 1 \end{matrix}$$

(b) $h_{1,0,1}(x,y,z) = h_{1,0}(x,y).h_1(z)$

$$= \begin{matrix} 1 & -1 \\ 0 & 0 \end{matrix} \cdot h_1(z) \qquad \text{from (ii) above}$$

$$= \begin{matrix} 1 & -1 \\ 0 & 0 \end{matrix} \quad \begin{matrix} -1 & 1 \\ 0 & 0 \end{matrix}$$

(c) $h_{1,0,1}(x,y,z) = h_{101}(x) = h_5(x)$

$$= \begin{matrix} 0 & 0 \\ 1 & -1 \end{matrix} \quad \begin{matrix} 0 & 0 \\ 0 & 0 \end{matrix}$$

Also, new types of one-dimensional Haar functions can be formed by writing out the rows or columns of a higher-dimensional Haar function.

Example:

From (i) above,

$$h_{1,2}(x,y) = \begin{matrix} 1 & 1 & -1 & -1 \\ -1 & -1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{matrix}$$

and so $h_6(x) = 1\ 1\ -1\ -1\ -1\ -1\ 1\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0$

whereas, according to the original definition,

$$h_6(x) = 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ -1\ -1\ 0\ 0\ 0\ 0$$

Relationship to Walsh functions

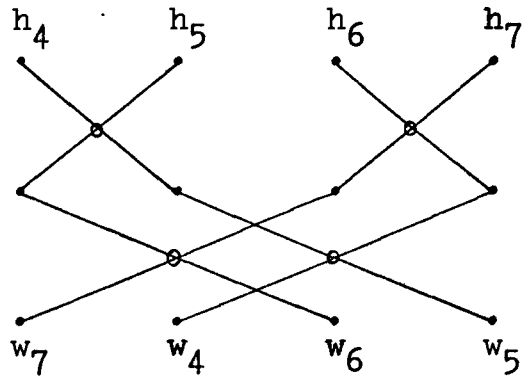
$$\begin{aligned} h_4(x) &= 1\ -1\ 0\ 0\ 0\ 0\ 0\ 0 \\ h_5(x) &= 0\ 0\ 1\ -1\ 0\ 0\ 0\ 0 \\ h_6(x) &= 0\ 0\ 0\ 0\ 1\ -1\ 0\ 0 \\ h_7(x) &= 0\ 0\ 0\ 0\ 0\ 0\ 1\ -1 \end{aligned}$$

$$\begin{aligned} w_4(x) &= h_4(x) - h_5(x) + h_6(x) + h_7(x) \\ w_5(x) &= h_4(x) - h_5(x) - h_6(x) + h_7(x) \\ w_6(x) &= h_4(x) + h_5(x) - h_6(x) - h_7(x) \\ w_7(x) &= h_4(x) + h_5(x) + h_6(x) + h_7(x) \end{aligned}$$

The pattern of additions and subtractions by which the Haar functions are combined to form the Walsh functions

is identical to that used in the fast Walsh transform to convert data into Walsh coefficients, i.e. it is the pattern of +1s and -1s representing the values of the Walsh functions themselves.

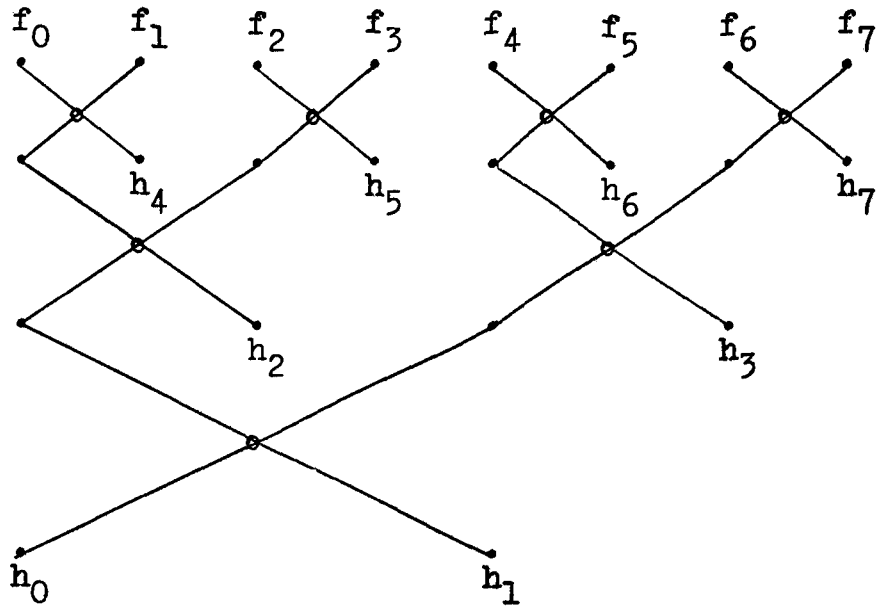
In the diagrammatic notation of Chapter 2, the above relationship may be shown as follows:




The fast Haar transform

It follows from the relationship between the Haar functions and the Walsh functions as shown in the previous section that the fast Haar transform is a part of the fast Walsh transform. The portion which is omitted is that part which would convert the Haar functions into Walsh functions, and which would therefore also convert Haar coefficients into Walsh coefficients.

The diagrammatic representation of the fast Haar transform is the difference between that for the fast Walsh transform, given on page36, and the diagram given above:



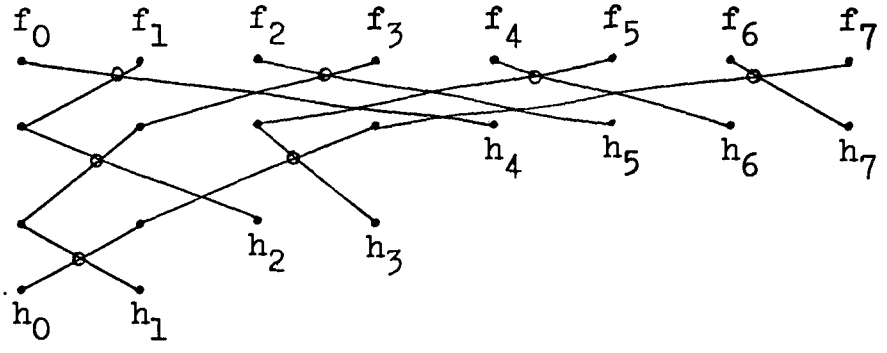

The data is represented by $f_0, f_1, \ldots, f_7$ and the points in the calculation at which the Haar coefficients are produced are indicated by $h_0, h_1, \ldots, h_7$.

Where there are $2^n$ data, the number of additions and subtractions involved is:

$$2^n + 2^{n-1} + \ldots + 2 = 2.(2^n - 1)$$

compared with $n.2^n$ for the fast Walsh transform.

The fast Haar transform can be modified so that each stage of the process is identical in structure to the others, although the scope of the computation is halved at each successive stage:



In this case, the coefficients are produced in the equivalent of the W ordering.

There is no simple equivalent for the Haar functions to the logical transform for the Walsh functions, because the logical transform depends upon the Walsh functions assuming the values +1 and -1 with near equal frequencies, and no other values.

The Haar functions have been used in signal processing, (10), but the fast Haar transform has only recently been used by other workers.

Orthogonal sets of order 4m

Paley (21) conjectured, but was unable to prove, that a square orthogonal matrix can be constructed, all of whose elements are +1 or -1 , provided that the matrix is of size n x n where n is of the form 4m , where m is a positive integer; except where n = 1, 2.

Such orthogonal matrices in effect define sets of orthogonal binary valued functions, in the way that Walsh matrices define the Walsh functions. It is sometimes inconvenient to use sets of functions of order $2^n$, especially in any type of approximation work, because of the low density of values of $2^n$ amongst the integers. It is therefore of interest that sets of order 4m can generally be constructed.

Although a computer program was written to find all possible sets of orthogonal binary valued functions of

order $4m$, its scope was limited by lack of computer space and time, and it produced no hitherto unknown sets of functions.

Particularly valuable are those sets of functions, of order $4m$, which are cyclic. A possible application for such functions is discussed in the following chapter.

# CHAPTER 6

# THE ANALYSIS OF SHAPES

## Pattern recognition

In any pattern recognition system there are normally three stages: input of data, transformation, discrimination. The transformation of the input data serves the purpose of providing a description on the basis of which discrimination, or recognition, can take place. It is simply the process of presenting the information in a suitable form for the discriminant mechanism. Complete systems of artificial pattern recognition can only hope to rival human performance when they are enhanced by reasoning and linguistic capabilities of a high order. In the absence of these techniques at the required levels, this chapter concerns itself solely with the first stage of the transformation process, and the place that analysis by means of Walsh functions might have in that process. (Problems associated with the input of data are discussed in Chapter 9).

Certain problems in the field of pattern recognition have already been tackled with considerable degrees of success. An obvious example is character recognition. Even handwritten characters can be successfully recognised given sufficient time. This is a simple problem in the sense that the domain and range of the patterns are known, well-defined and relatively restricted.

The complexity of, say, giving a description of a general scene as viewed by a robot through a television camera 'eye' is clearly much greater. In such a situation, a general purpose pattern recognition system would be more suitable than a collection of restricted systems, such as those for character recognition or chromosome analysis; and would rely heavily upon learning abilities.

The methods described here are not special purpose, nor do they claim to be more than possibly a small contribution to a general purpose system. They might, however, find much wider and more immediate applicability in circumstances under which the final stage of recognition is not required. For example, in a study of leaf shapes, such as that cited in Chapter 8.

Shape analysis

Assume that the data is a digitised representation of a two-dimensional shape; that the representation is to be transformed into a set of values to which the Walsh transform can be applied; and that a spectrum is to be formed and interpreted in a spatially meaningful sense.

An optical image is normally presented to a digital computer as an array, or retina, of regular cells. Associated with each cell is a digitised value, which may correspond to the intensity of the original image in that area, for example, or may represent a colour code, etc.

This set of values is the retinal function. The cells may be square or hexagonal. The latter will generally give a better fit to the original image, but for conceptual reasons it is easier to discuss the processing of retinal functions in terms of the former.

The logical Walsh transform was designed specifically to deal with binary data of this form.

Walsh functions, and other sets of orthogonal functions, are useful tools in the approximation of given functions. In the case of Walsh functions there are two particular advantages over most other methods of dealing with visual input. Firstly, the amount of information to be processed is likely to be large, and it is desirable to have methods of computation which can be carried out in parallel. This is possible with both the fast Walsh transform and with the logical transform. Secondly, the spectrum of the Walsh coefficients can be interpreted as a decomposition of the shape into its spatial components, and spatial filtering techniques can be used.

Two-dimensional shapes

A two-dimensional Walsh transform can be applied directly to a retinal image. This normally involves a prohibitive amount of information processing, but if the retinal function is binary valued (i.e. there are only two grey levels - black and white) it becomes a feasible proposition if the logical transform is used, and the data is in fact treated as one-dimensional.

For retinal functions which assume several values, corresponding to, say, eight grey levels, a solution depends upon the development of faster hardware; but for binary valued retinal functions there are, in any case, alternative methods which will be discussed later in this chapter.

Assuming for the moment that the retinal function consists of N x N square cells, associated with each of which is a value - 1 or 0 , and that this data is stored in an array D , whose typical element is denoted by $d_{i,j}$ : This array can be represented exactly by a set of two-dimensional Walsh coefficients. Note that the only approximation to the original image is at the input stage, when it is digitised on a cellular array. The Walsh transform involves no loss of information.

D can be expressed as a linear combination of N x N Walsh functions:

$$D = c_{0,0} \cdot w_{0,0} + c_{0,1} \cdot w_{0,1} + \cdot \cdot \cdot + c_{N,N} \cdot w_{N,N}$$

where $c_{i,j}$ is the Walsh coefficient representing the correlation between D and the Walsh function, $w_{i,j}$.

The coefficients are computed by means of the Walsh transform. The logical transform is most suitable since the data is binary valued.

If the data represents a shape whose cells are connected there are two short cuts which can be taken in the computation of a complete specification of the shape in terms of Walsh coefficients:

The first is that the coefficients $c_{i,j}$ for which $i \geqslant N/2$ and $j \geqslant N/2$ can be deduced from those for which $i < N/2$ or $j < N/2$. In other words, only three-quarters of the coefficients are required. It is equivalent to saying that the shapes of Figs. 1 and 2 are the same.

The second short cut is to regard the shapes of Figs. 3 and 4 as being the same, and the sets of coefficients which specify them as being equivalent.

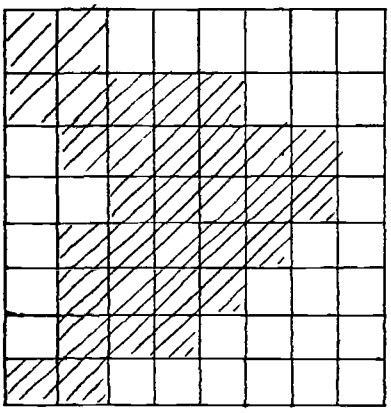


Fig. 1

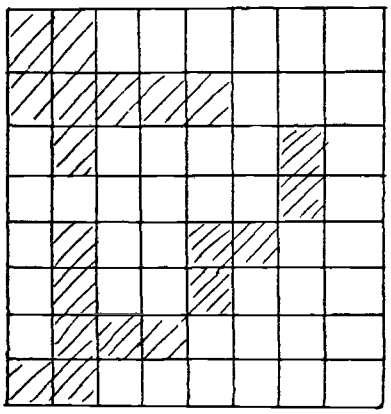


Fig. 2

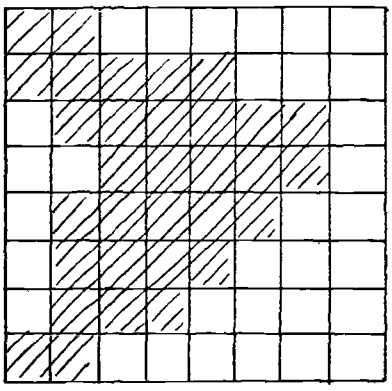


Fig. 3

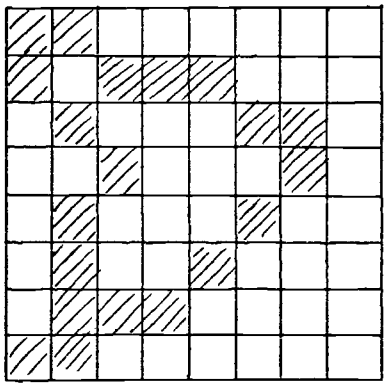


Fig. 4

Spectra

If a one-dimensional Walsh spectra is to be composed from a set of two-dimensional coefficients - notwithstanding that they have been produced by a one-dimensional transform - an appropriate mapping from the two indices of the coefficients to a single index must be devised. $c_{i,j}$ is the coefficient corresponding to the Walsh function, $w_{i,j}(x,y)$, in the W ordering. Since $w_i(x)$ has i discontinuities and $w_j(y)$ has j discontinuities, $w_{i,j}(x,y)$ will have $N.(i+j)$ discontinuities if they are counted along each row and each column of the array of values representing the function, $w_{i,j}(x,y)$.

Example:

$$w_{2,3}(x,y) = \begin{array}{cccc} +1 & -1 & +1 & -1 \\ -1 & +1 & -1 & +1 \\ -1 & +1 & -1 & +1 \\ +1 & -1 & +1 & -1 \\ 2 & 2 & 2 & 2 \end{array} \quad \begin{array}{l} 3 \text{ discontinuities} \\ 3 \\ 3 \\ 3 \end{array}$$

Total = $4.(2+3) = 20$

Therefore, the total number of discontinuities is proportional to $i+j$ and, by analogy with the one-dimensional case, $c^2_{i,j}$ is plotted against $\frac{1}{i+j}$. Normally there will be several coefficients the sum of whose indices will equal a given integer value, eg. $c_{3,2}$ and $c_{4,1}$. In these cases, the average squared coefficient (e.g. $\frac{c^2_{1,1} + c^2_{2,0} + c^2_{0,2}}{3}$) is plotted against $\frac{1}{i+j}$.

The techniques of spatial filtering, in which only selected coefficients are calculated or plotted in the spectrum, can be applied as readily to the two-dimensional case as to the one-dimensional.

Standardisation

Before going on to consider the question of transforming two-dimensional data to one-dimensional in an attempt to cut down the computation involved in applying a Walsh transform, the problem of shape standardisation will be discussed.

An obvious problem is that the intuitive concept of shape is unaffected by changes of the retinal function due to size change, translation or rotation. If these are not taken into account, then the same 'shape' may on different occasions produce different sets of Walsh coefficients.

These difficulties do present formidable obstacles when using either Walsh functions or the sinusoidal functions, (18). Some writers have dealt with these difficulites in purely theoretical terms, applicable only to infinitely small cellular arrays.

A possible approach to the problem is to define some type of more general Walsh coefficient. For example, one might effect all possible transformations (size changes, rotations and translations) of a particular shape, compute the Walsh coefficients for each transformation, and select the algebraically largest $c_i$ for each $i$.

If the set of all transformations is denoted by $T$, then the shape would be specified by a set of coefficients $C'$ where

$$c'_{i,j} = \max_{T} c_{i,j}$$

Alternatively, one might take $c'_{i,j}$ to be the average squared coefficient over all transformations:

$$c'_{i,j} = \sum_{T} c^2_{i,j}$$

Note: if the coefficients were not squared before averaging, the result would be zero for all $c'_{i,j}$ except $c'_{0,0}$.

Counter-examples have been found which show that neither of the above definitions of general Walsh coefficients provide a unique description of a shape. In any case, this type of method involves an enormous amount of computation, unless the set of coefficients $C'$ can be found directly from any set of coefficients corresponding to one of the transformations belonging to $T$. This 'short cut' is not apparently easily attainable.

The alternative to a general Walsh coefficient is to devise a procedure for standardising a shape with regard to size, rotation and translation.

With regard to size, it is quite simple to reduce or increase the original image by optical means so that a fixed number of the cells of the retinal function will have the value $+1$.

Translation can be standardised upon by fixing the co-ordinates in the cellular array of the centre of gravity of the shape. This can again be done optically by finding the centre of intensity or brightness of the original image. The problem of standardising upon rotation and therefore upon the complete shape appears impossible, however:

> " . . . the problem of centering and of normalising the dimensions of figures generates a vicious circle: in order to recognise a figure it should be transformed, but in order to effect correct transformation the figure should first be recognised. . . . "
>
> (Teaching computers to recognise patterns: A. G. Arkadev & E. M. Braverman, Academic Press, (London and New York) 1967 )

It is in fact the major unsolved problem of pattern recognition theory. Special cases, such as character recognition, are amenable to solution since the data is presented in approximately standardised form with respect to rotation, e.g. the characters are roughly 'upright'. In particular situations, use may be made of contextual information, or a standardisation procedure of limited scope may be arbitrarily fixed for a set of shapes which are to be compared. There are also methods which may work for a majority of cases, but which fail in the face of ambiguities or certain types of symmetry. For example, one might standardise upon rotation by fixing the alignment of the longest axis of the shape - an axis being

defined as any line joining two points on the boundary of the shape. This is a most unsophisticated technique, with obvious pitfalls.

Particularly if shapes are to be dealt with by the application of a Walsh transform to the input data, then further investigation of the standardisation problem in the general case is required.

Reduction of data to one-dimensional form

It is assumed that the retinal function takes only the values of 'black' and 'white' - 1 and 0 ; that the function represents a shape, which is defined as the interior of a simple closed curve or a set of connected cells having the value 1 .

A square cell has as neighbours the four cells which have edges in common with it. A boundary corner is a corner of a cell of value 1 which is also a corner of a cell of value 0 . A boundary cell has value 1 and at least two boundary corners. These definitions are consistent with normal intuition.

A procedure whereby the two-dimensional retinal function can be transformed into a one-dimensional function is described:

The boundary cells of a retinal function can be found by means of an edge-following algorithm, or by a straightforward parallel computation which produces for each cell the value of the following expression:

if C = 1 and C1 + C2 + C3 + C4 < 4 then 1 else 0

where C is the value of the cell under consideration and C1, C2, C3 and C4 are the values of its neighbours.

The result 1 indicates that the cell having value C is in fact a boundary cell, the result 0 that it is not. The form of the function follows directly from the alternative definition of a boundary cell as one which has value 1 and at least one neighbour having value 0 . In the case where all four neighbours have value 0 , the central cell is a complete shape in itself.

The centre of gravity of the shape can be found optically or digitally, as the point whose co-ordinates are the averages of the co-ordinates of all the cells having value 1 .

The distances between the centre of gravity and the boundary corners are calculated, beginning at an arbitrary corner and proceeding clockwise or anti-clockwise according to a fixed convention. Note that the boundary corners are equally spaced along the perimeter of the shape.

Suppose the distances from the centre of gravity to the boundary corners are $g_1, g_2, \ldots, g_m$ . Firstly, these values are approximated to by a set of values $f_1, f_2, \ldots, f_N$ , where N is of the form $2^n$, as on pages 25 and 26, and N is the largest such number smaller than m .

To minimise the information loss at this stage, it is desirable that the original image should be reduced or increased in size so that $m$ is only slightly larger than a value of $2^n$.

The set of values $f_1, f_2, \ldots, f_N$ are scaled so that their average value equals a fixed constant. This is equivalent to standardisation upon size. By taking measurements from a point fixed relative to the shape - the centre of gravity - translation has also been stand ardised upon.

The fast Walsh transform is applied to the scaled set of values to produce $N$ coefficients. The logical transform may be used, but each datum is no longer a single bit, since it represents a measurement from the centre of gravity to a boundary corner.

The spectrum formed from the one-dimensional coeff-icients is spatially meaningful, although quite different from that formed from two-dimensional coefficients for the same shape, since the boundary cells and therefore the distances from them to the centre of gravity are a complete specification of the shape.

A slight modification must be made in the construction of a spectrum when the above method is used. The Walsh functions, $w_{2m-1}(x)$, $m = 1,2, \ldots$, have an extra discontinuity between the initial and final values. When correlated with cyclical data (as here, where the last datum and first datum correspond to adjacent boundary corners) the spectrum is formed by plotting $\frac{c_{2m}^2 + c_{2m-1}^2}{2}$ against $\frac{1}{2m}$ for $m = 1, 2, \ldots, \frac{N-2}{2}$

The Walsh transform decomposes the shape into its spatial components, and the spectrum represents this decomposition visually. The high frequency part of the spectrum represents the amount of fine structure of the shape, and the low frequency represents the coarse structure.

Examples will be found in Chapter 9 - the analysis of leaf shapes.

The one disadvantage of this method of reduction from the two-dimensional retinal function to a one-dimensional set of data is that in very rare cases the set of values representing the distances from the boundary corners to the centre of gravity could be used to construct two or more different shapes. In this sense, the distances do not uniquely specify the retinal function; although it is emphasised that ambiguities occur with an almost insignificant frequency.

Cyclical functions

Although the method of the previous section standardised upon size and translation, the problem of standardising upon rotation has still to be dealt with. It can be partially overcome by using a set of cyclical boolean orthogonal functions instead of the Walsh functions. Example:

$$f_0(x) = \quad 1 \quad 1 \quad -1 \quad 1$$
$$f_1(x) = \quad 1 \quad 1 \quad 1 \quad -1$$
$$f_2(x) = \quad -1 \quad 1 \quad 1 \quad 1$$
$$f_3(x) = \quad 1 \quad -1 \quad 1 \quad 1$$

The typical function in such a set is a digital sequence with pseudonoise properties. The set of Fourier coefficients corresponding to such a set of functions is unique for a shape standardised by the above methods. If a standard rotational position is somehow fixed (i.e. if a criterion can be established for selecting the first boundary corner from which to measure the distance to the centre of gravity), and the resulting set of coefficients is

$$c_0 \quad c_1 \quad c_2 \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad c_N$$

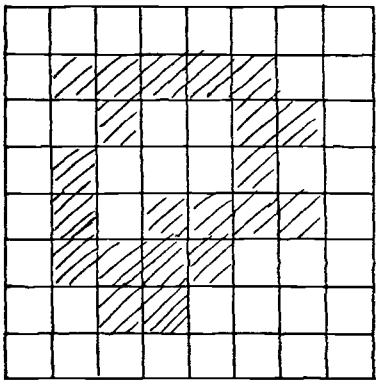
then, in general, the same shape will produce coefficients:

$$c_k \quad c_{k+1} \quad \cdot \quad \cdot \quad c_N \quad c_0 \quad c_1 \quad \cdot \quad \cdot \quad c_{k-1}$$

Such sets of cyclical functions do not have the spatially interpretative properties of the Walsh functions.

Patterns of shapes

The 'shape' shown in the data array below is not a set of connected cells of value 1 . Its analysis could be carried out by means of the two-dimensional techniques described at the beginning of this chapter, but the resulting spectrum would be difficult to interpret.



Also, the data cannot be reduced to a single dimension because the boundary corners are not in any definable order which would be meaningful with respect to the 'shape'. It is therefore necessary to regard the figure as a 'pattern' of shapes; in this particular case, as a black shape with a white shape superimposed upon it, or as a black shape with a 'hole' in it.

There are several methods by which the pattern of shapes can be decomposed into the constituent shapes. Unfortunately, although the task of decomposition can be carried out in a single scan of the cellular array (1), it cannot be carried out by means of a computation executed in a parallel mode, as shown by Minsky and Papert in an equivalent theorem (19, page 74).

With the pattern decomposed into its constituent shapes, each is analysed by finding the centre of gravity, boundary corners, etc. The only difference between the normal procedure and that for patterns of shapes is that in the latter case the distances from boundary corners to centres of gravity are standardised not by scaling their average to a fixed constant value, but by scaling them in proportion to the areas of the corresponding shapes.

Haar functions

A simple connected binary valued shape can also be defined by a pair of ternary valued one-dimensional sets of data.

One of the pair records the change in the y co-ordinate as the outline is followed from boundary corner to boundary corner, and the other records the change in the x co-ordinate. Each may take any of the values 0, +1 and -1 . These function values could be produced by an edge following algorithm ( an example of which appears below ) and the analysis might be in terms of the Haar functions, since they also take the values 0, +1 and -1 .

Edge follower

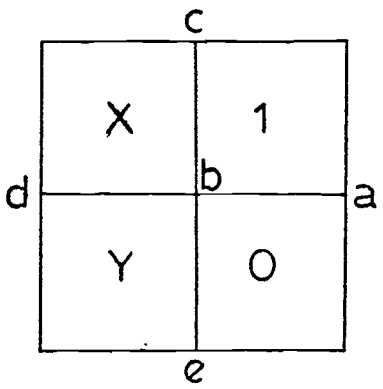
The cellular array is C .

```
begin
read(startx, starty)      : the co-ords. of the first corner
k = 0                     : counts the number of corners
i = startx                : i and j are the co-ords. of
j = starty                  the current corner
1:check                   : checks whether the follower
                            is back at the starting point
test                      : looks at corners i,j+1 , i,j-1 ,
                            i+1,j and i-1,j (excluding the
                            previous corner) to see which is
                            a boundary corner
end
```

check: if i = startx and j = starty and k ≠ 0 then stop

test: see diagram on next page.

The general situation can be summarised by the diagram below:




a is the previous boundary corner

b is the current boundary corner

the cells have the values indicated

if the cell X has the value 0 then c is the next boundary corner; if it has the value 1 then d is the next boundary corner, unless Y also has the value 1, in which case e is the next boundary corner.

# CHAPTER 7

## ORTHOGONAL FUNCTIONS IN PATTERN RECOGNITION

Recently there has been a considerable increase in the use of orthogonal functions in pattern recognition and shape description work of one type and another. In this chapter, brief details are given of such work and the techniques used are compared with those which might be employed using Walsh function.

On a historical note, J. J. Sylvester used positive and negative signs to describe patterns in colour, tile work, etc, in 1867. However, his studies amounted to little without the power of the digital computer, so suited to his binary representation.

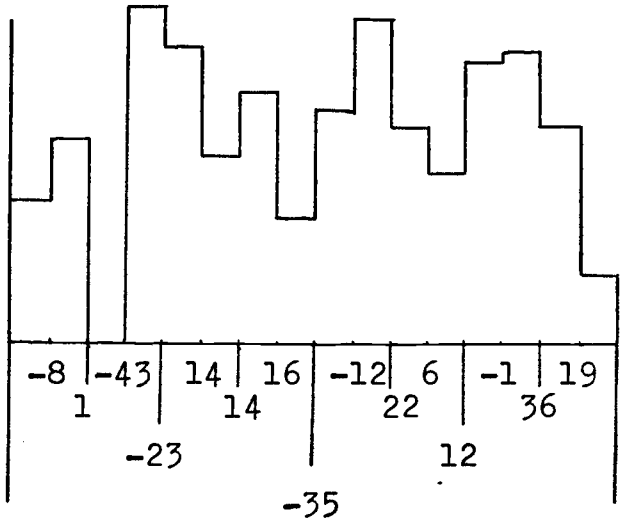
### Haar functions

Gerardin and Flament (10) have used Haar functions in what they refer to as 'geometrical pattern feature extraction'. The analysed pattern may take only the values 0 or $h$, the value of $h$ being such that the integral of the function (or pattern) is unity over the given interval. The pattern is correlated with the Haar functions to produce Haar coefficients. Each coefficient is associated only with that part of the interval for which the corresponding Haar function is non-zero.

Example:

non-binary data

$f_k$




Haar coefficients

They then state interpretation rules of the following type:

"A positive coefficient signifies that on the first half of the interval the signal is longer than on the second half; this indicates that at least one discontinuity occurs in the interval. A negative coefficient leads to the inverse conclusion."

and

"If all the coefficients from $c_{2^{m-2}}$ to $c_{2^{m-1}}$ are zero and if the sum of the squares of the coefficients calculated up to then is equal to the energy contained in the signal, all the other coefficients will be zero and the coefficients calculated exactly sum up the pattern."

Gerardin and Flament did not use the fast Haar transform; they did not use a one-dimensional representation of two-dimensional Haar functions so that a one-dimensional transform could be used. As they only worked with binary data (taking the values 0 and h) and were frequently concerned with whether coefficients were

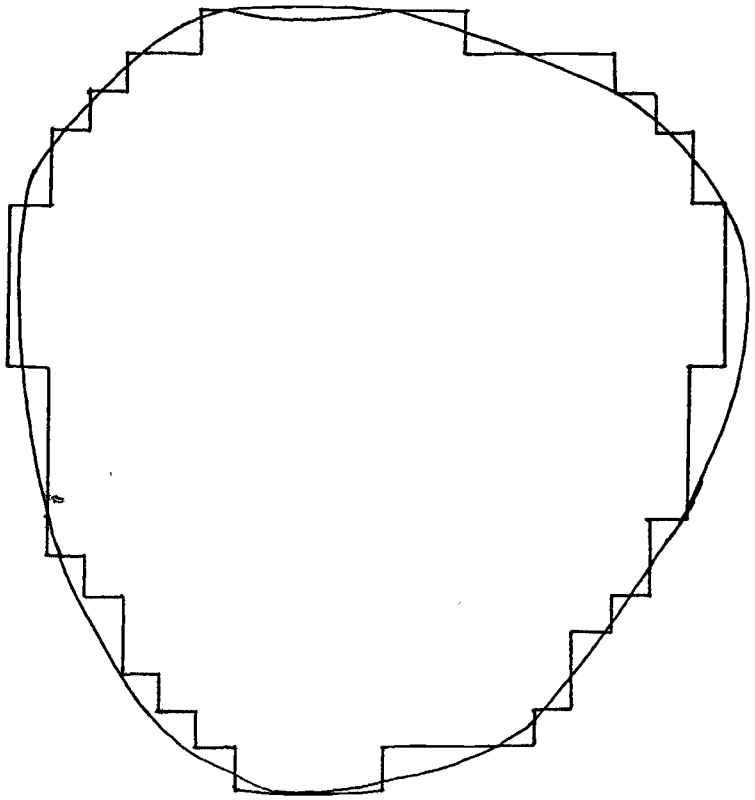
positive or negative, their problem might well be solved by using the logical Walsh transform.

The mapping of two-dimensional data onto one dimension was dealt with at length in the previous chapter. In another way, it has also been considered by Lu and Rutowitz.

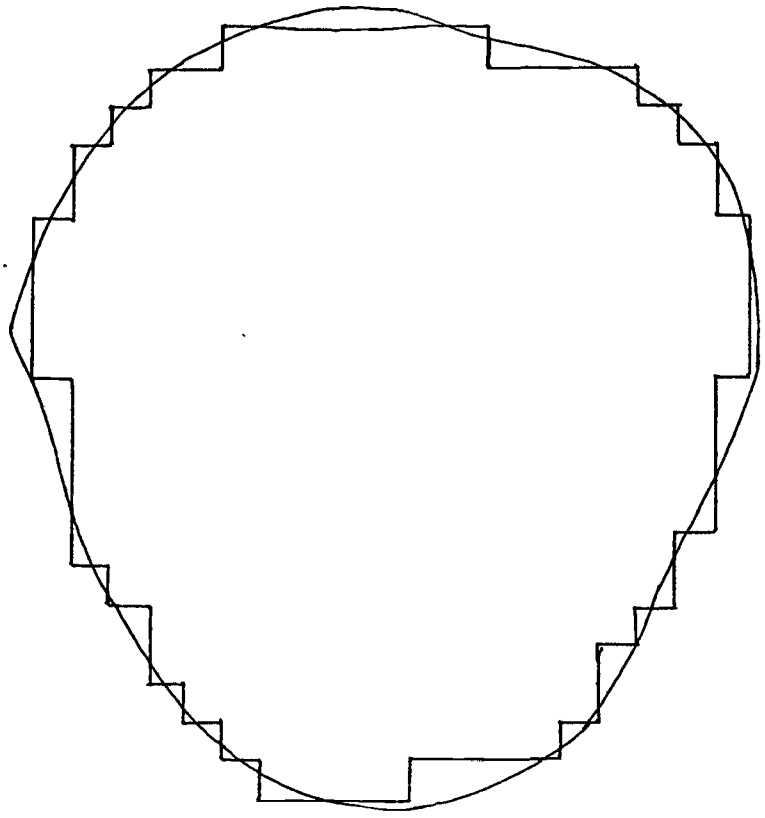
Faces and chromosomes

Lu's original data is an outline drawing of the human face, head or skull (15). The contextual information provided by such features as the eyes, nose, etc., are used to fix the translation (vertical and horizontal) and rotation of the shape. Then, in much the same way as the method of the previous chapter, measurements are taken from a fixed origin at the centre of the shape to the outline, not at regular intervals along the perimeter, but at regular angular intervals.

The data is then analysed in terms of Fourier coefficients for the sinusoidal functions, although Lu does not take advantage of the fast Fourier transform. It might be thought that the curvature of the outline of such a shape would lend itself to approximation by the sine and cosine functions, but the Walsh functions are better approximators to the function which results from the measurements to the outline from the origin; as the example below shows, fewer coefficients are needed for similar accuracy when the Walsh functions are used.



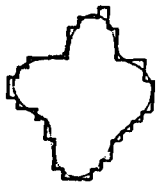
Approximation to digitised skull outline by 16 Walsh coeffs.



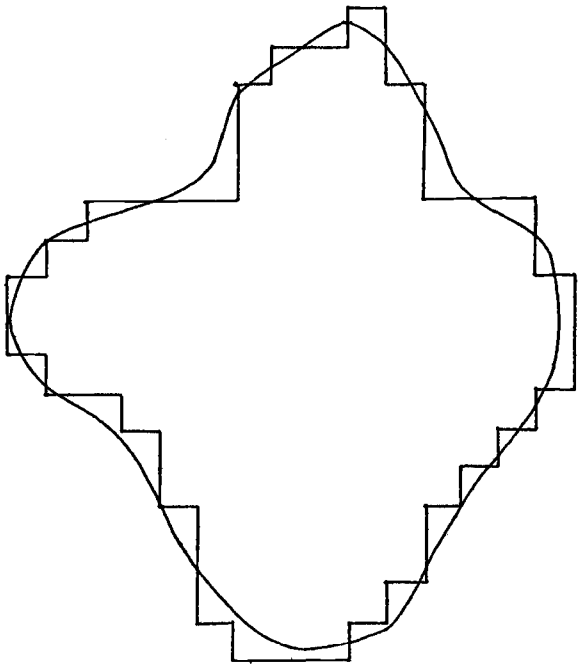
Approximation to digitised skull outline by 13 Fourier coeffs.

Rutowitz's work with chromosome outlines employs a method which is substantially the same. The shape outlines are defined by a set of polar co-ordinates. The origin is not the centroid of the outline, but the point about which the first Fourier coefficient is zero. This point is found by an iterative procedure. (The technique of using an origin about which the Walsh coefficient $c_1$ is zero can be used, but contributes nothing towards the solution of the standardisation problem). The possibility of classifying chromosomes into their different types on a basis of the analysis of their outlines in terms of a few Fourier coefficients is obstructed by the apparent insolubility of the standardisation of rotation problem. However, Rutowitz claims that only Fourier coefficients up to the sixth order are required to give an excellent approximation to the chromosome outlines. Of course, the calculations are much greater than those necessary if the logical transform, or even the fast Walsh transform, were used.

The results below show what can be achieved using Walsh functions compared with the sinusoidal functions. The example is taken from Rutowitz's paper, 'Centromere finding : some shape descriptors for small chromosome outlines', Machine Intelligence 5, (eds. B. Meltzer & D. Michie), Edinburgh University Press, 1970.



Approximation by sinusoidal functions.
All terms to the sixth order.
(13 coefficients)



Approximation by Walsh functions.
All terms to the fourth
order. (16 coefficients).

Spatial filtering

Tallman establishes that selective spatial filtering of Fourier images of visual stimuli tends to cluster similar patterns and permits their accurate classification, (28). His method is to apply the fast Fourier transform to a two-dimensional function of Os and 1s which represents the shape of an alphabetical character; then simply eliminating the high spatial frequency components to the point where further exclusion of information would decrease the probability of accurate classification due to lack of information, and where to have included more frequency components would have increased the spatial 'noise' content of the transform, and included information of no relevance to the problem of classification. Tallman suggests that it would be desirable if the inverse transform of a truncated transform produced a 'fuzzy' copy of the original image. This appears to be the case at least for the Walsh functions (see the conjecture of Chapter 4).

Carl shows that similar results can be obtained using the Walsh functions (2), although his method is somewhat different. He applies a two-dimensional fast Walsh transform (for which he has an algorithm which is equivalent to that for the one-dimensional fast Walsh transform) to a 64 x 64 raster of binary data. In such a case, the logical transform may be more appropriate. However, 64 x 64 Walsh coefficients, which are not binary valued are produced by the transform.

Discrimination between the various classes of image (again, alphabetical characters) takes place on the basis of these coefficients. Carl does not make a sharp distinction between high and low frequency coefficients, but by a learning algorithm selects those coefficients which are most useful in the process of recognition. It turns out in practice that they are low frequency coefficients. On different occasions, Carl uses 25, 49 or 81 coefficients in the discrimination process. Depending upon the accuracy with which a training set of images is classified on the basis of any of the above number of coefficients, the post-training work is done using a raster of dimensions $5 \times 5$, $7 \times 7$, or $9 \times 9$.

Feature spaces

Coombs uses 'caltrop' matrices in his pattern recognition system, (6, 7). Such a matrix is defined by the following conditions:

(i) Each element takes the value +1 or -1

(ii) The rows and columns must be orthogonal to one another.

(iii) The matrix is of order $4m$

(iv) Each row, except for the first, and each column, except for the first, contains $2m$ -1s and $2m$ +1s

(v) Rows and columns are multiplied by -1 where appropriate so that there is one complete row and one complete column of +1s. (This can be done without affecting the orthogonality).

Both the Walsh matrices and the modified Walsh matrices are examples of caltrop matrices.

Each column of a caltrop matrix represents a feature vector. Because of the requirements placed upon the form of the matrix, the feature vectors are equally spaced in the appropriate dimensional space.

Each row represents a set of features which are present (+1) or absent (-1) ; and a category is associated with each set of features.

The well-known Adaline algorithm is applied to a training set of characters so that actual features of the images can be automatically selected and identified with the elements of the feature vectors. After the training period, a character is classified according to the category which it most closely resembles.

Example:

Caltrop matrix

| categories | features | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 1 | +1 | +1 | +1 | +1 | +1 | +1 | +1 | +1 |
| 2 | +1 | +1 | +1 | +1 | -1 | -1 | -1 | -1 |
| 3 | +1 | +1 | -1 | -1 | -1 | -1 | +1 | +1 |
| 4 | +1 | +1 | -1 | -1 | +1 | +1 | -1 | -1 |
| 5 | +1 | -1 | -1 | +1 | +1 | -1 | -1 | +1 |
| 6 | +1 | -1 | -1 | +1 | -1 | +1 | +1 | -1 |
| 7 | +1 | -1 | +1 | -1 | -1 | +1 | -1 | +1 |
| 8 | +1 | -1 | +1 | -1 | +1 | -1 | +1 | -1 |

If a character has features 1, 4 and 5 , but not 7, 2, 3, 6 and 8 , it will be classified as category 5, since it more closely resembles the features which are absent and present in category 5 than in any other category. This may be expressed by saying that the Hamming distance between the character and the feature vector of category 5 is 1 , and that between any two categories is 4 . The Hamming distance is defined as the number of elements (+1s and -1s) in which the two arguments differ.

Although it is possible to measure the correlation between a character and a category's features, it is only necessary to use the knowledge of whether a character has a given feature or not, i,e. whether the correlation is positive or negative - the degree of correlation is not required. (Coombs has suggested that a third, 'don't know' alternative would be useful).

The system has much in common with the work of this thesis, particularly in its use of orthogonal boolean function matrices, and its use of negative and positive correlation between a character and the elements of a feature vector. The link with the logical Walsh transform is that the logical transform uses features which are no more than a single square of the two-dimensional input array, whereas Coombs' method constructs more complex features, by means of the Adaline algorithm.

Coombs' system, which is designed for application to handwritten, alphanumeric characters, is more sophisticated

and accurate than that of, say, Carl (2), who applies a Walsh transform directly to the data. The latter's system, however, is much faster, and efficient.

The search for the optimal caltrop matrix, given a training set of patterns, is similar to the search for an ideal set of orthogonal boolean functions, which was mentioned in Chapter 5, and has also been remarked upon by both Carl and Tallman (28).

# CHAPTER 8

## LEAF SHAPES : WALSH ANALYSIS

The leaf shapes of Fig. 1 show sixteen chrysanthemum leaves grown under varying conditions. There were four plants - a, b, c and d - grown under the following combinations of temperature and day length:

| | short day | long day |
|---|---|---|
| low temperature | a | c |
| high temperature | b | d |

The numbers - 1, 2, 3 and 4 - relate to the positions of the leaves of the plants. There are, therefore, four sets for comparison, each containing four leaves:

| | | | | |
|---|---|---|---|---|
| Set 1 : | a1 | b1 | c1 | d1 |
| Set 2 :: | a2 | b2 | c2 | d2 |
| Set 3 : | a3 | b3 | c3 | d3 |
| Set 4 : | a4 | b4 | c4 | d4 |

For each leaf shape, a point was defined by taking the midpoint of the line which most nearly split the leaf into two portions which were mirror images of one another. Using this point as an origin, distances were measured from it to points spaced at regular intervals along the boundaries. This method is more accurate than taking

measurements along radii of equal angular separation, since it is the outline which is to be approximated, and evenly spaced samples of the distance from the outline to the origin will achieve this with the greatest uniformity.

Consider the two following sets of data:

(a) $f_0$ $f_1$ $f_2$ $f_3$ $f_0$ $f_1$ $f_2$ $f_3$

(b) $f_0$ $f_1$ $f_2$ $f_3$ $f_3$ $f_2$ $f_1$ $f_0$

| Walsh functions | | | | | | | | in case (a) these coeff. will be 0 | in case (b) these coeff. will be 0 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | | |
| 1 | 1 | 1 | 1 | -1 | -1 | -1 | -1 | X | X |
| 1 | 1 | -1 | -1 | -1 | -1 | 1 | 1 | X | |
| 1 | 1 | -1 | -1 | 1 | 1 | -1 | -1 | | X |
| 1 | -1 | -1 | 1 | 1 | -1 | -1 | 1 | | |
| 1 | -1 | -1 | 1 | -1 | 1 | 1 | -1 | X | X |
| 1 | -1 | 1 | -1 | -1 | 1 | -1 | 1 | X | |
| 1 | -1 | 1 | -1 | 1 | -1 | 1 | -1 | | X |

If the origin of the leaf shapes has in fact been chosen so as to split the shape into two portions that are mirror images of one another, then the situation is as in (b) above, and the coefficients $c_1$, $c_3$, $c_5$, . . will be zero.

The spectrum will be of $\frac{c_{2m}^2 + c_{2m-1}^2}{2}$ plotted against $\frac{1}{2m}$ for the reasons explained on page 81. In practice, the $c_{2m-1}^2$ component will be small.

Distances were measured for each leaf from approximately 120 boundary points to the respective origins. From these distances, 64 data values were obtained as follows:

$$f_k = \frac{f_{max} + f_{min}}{2} \qquad \text{for } k = 1, 2, \ldots, 64$$

where $f_{max}$ and $f_{min}$ are the maximum and minimum of the distances which fall into the kth of 64 subintervals which the length of the boundary is divided into.

Sixty-four Walsh coefficients were computed for each leaf, and the spectra from $\frac{1}{32}$ to $\frac{1}{64}$ are shown in Fig. 2. The lower frequency end of the spectra are similar, due to the general similarity of the leaves, but the higher frequency end of the spectra shows the slightly less obvious variations in detail.

Other examples of the Walsh analysis of leaf shapes can be found in (17).

The rationale of this work is clearly stated in (16).

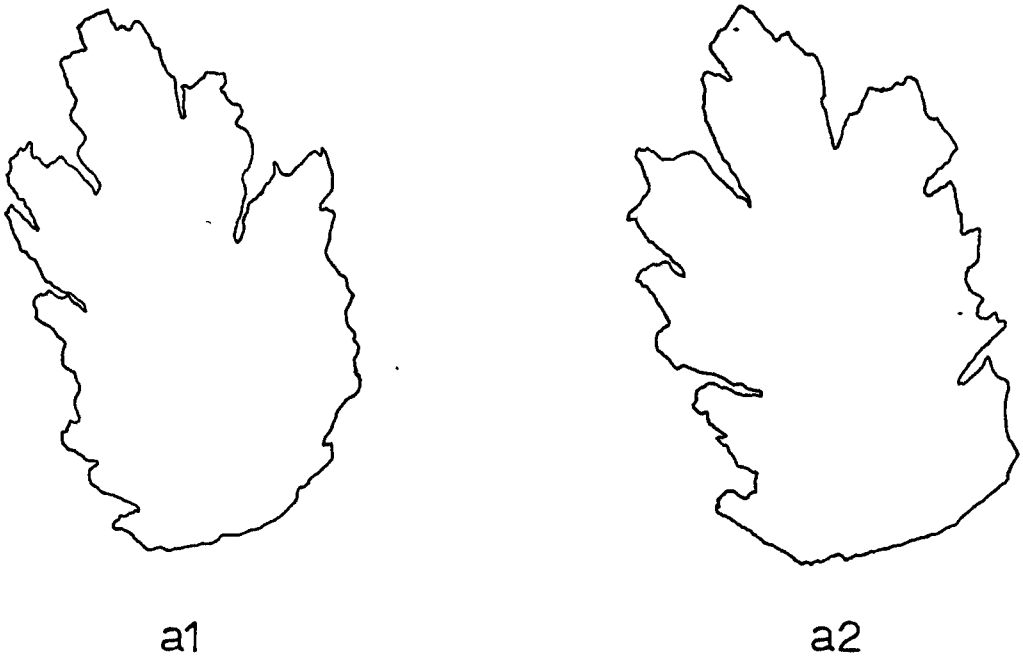
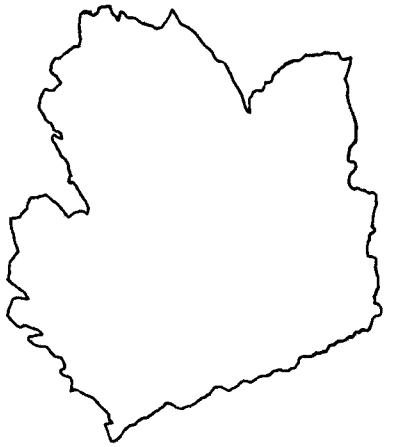
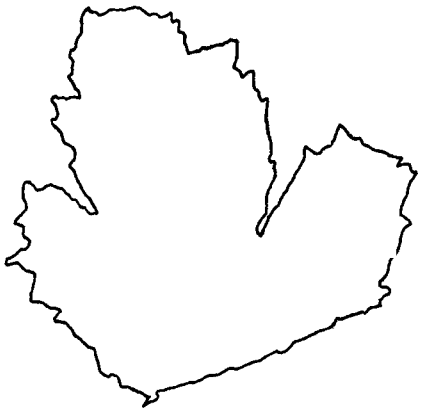


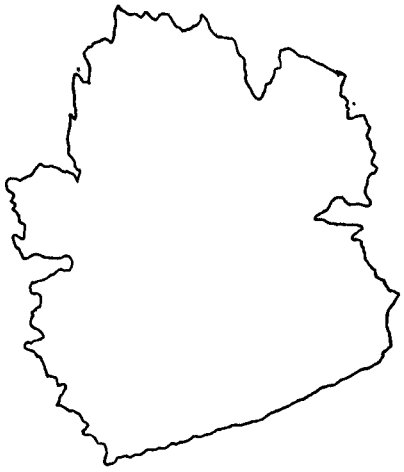

fig1



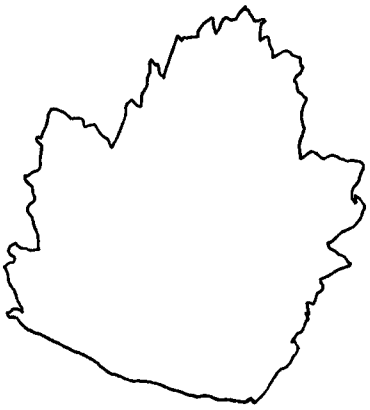
b1



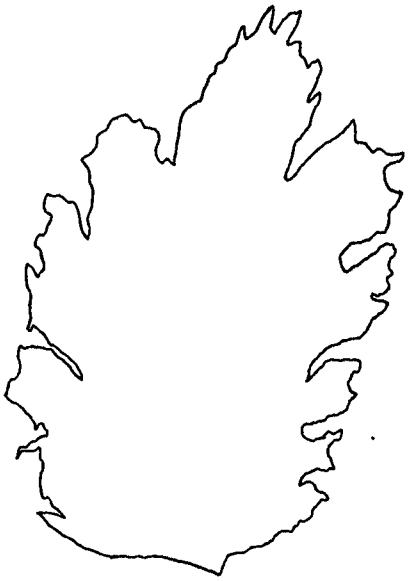
b2



b3

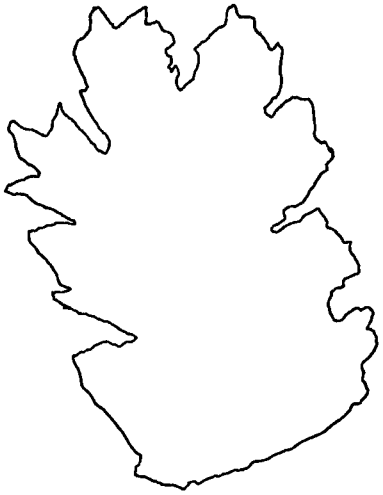


b4

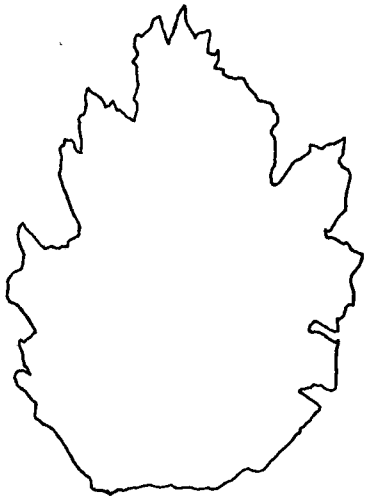


c1

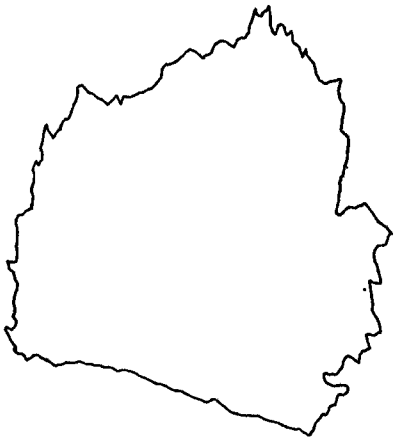
c2



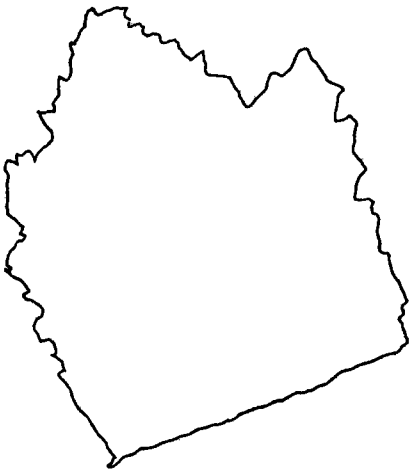
c3



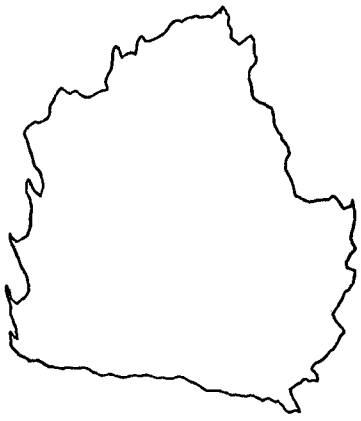
c4



d1

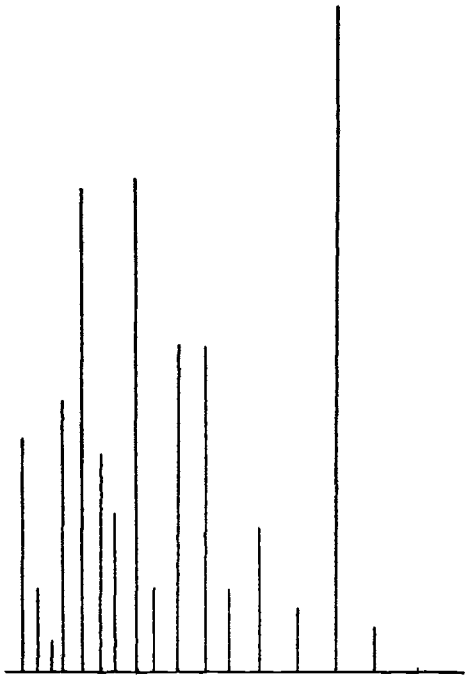


d2

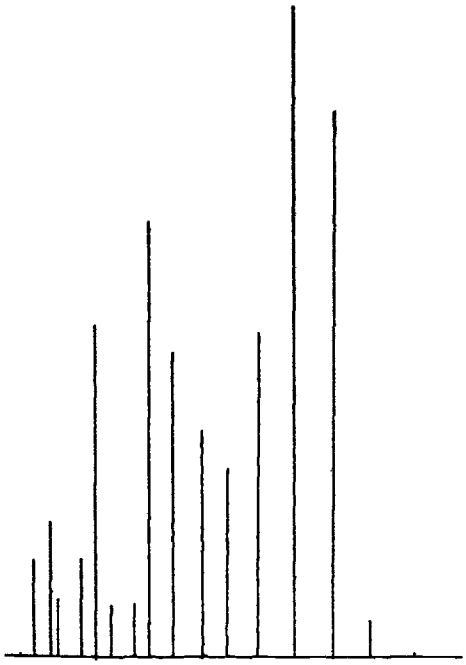


d3

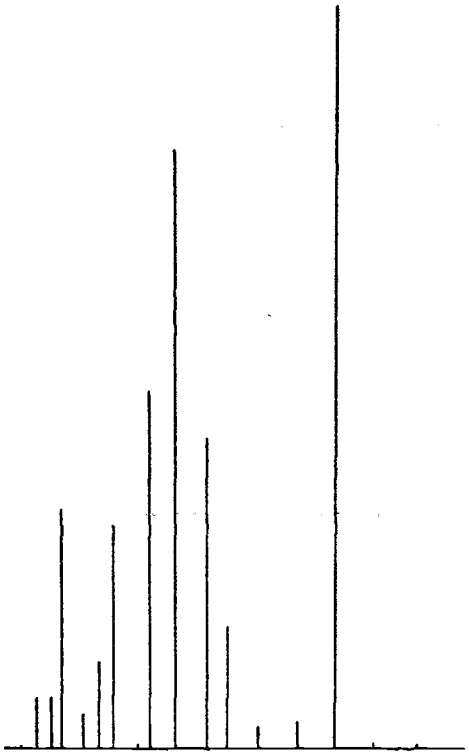
d4



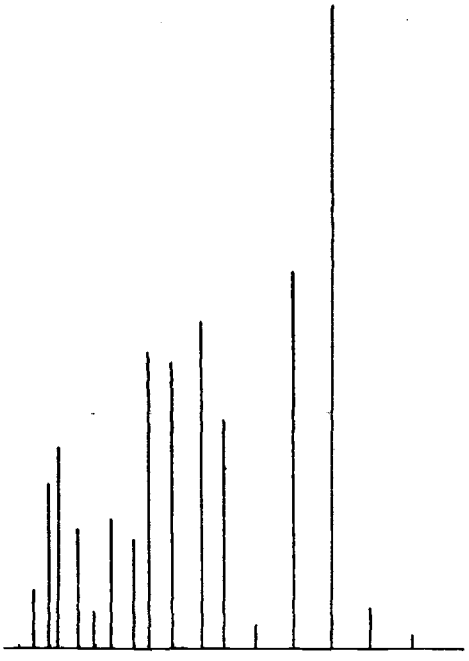
a1



a2



a3



a4

Fig 2

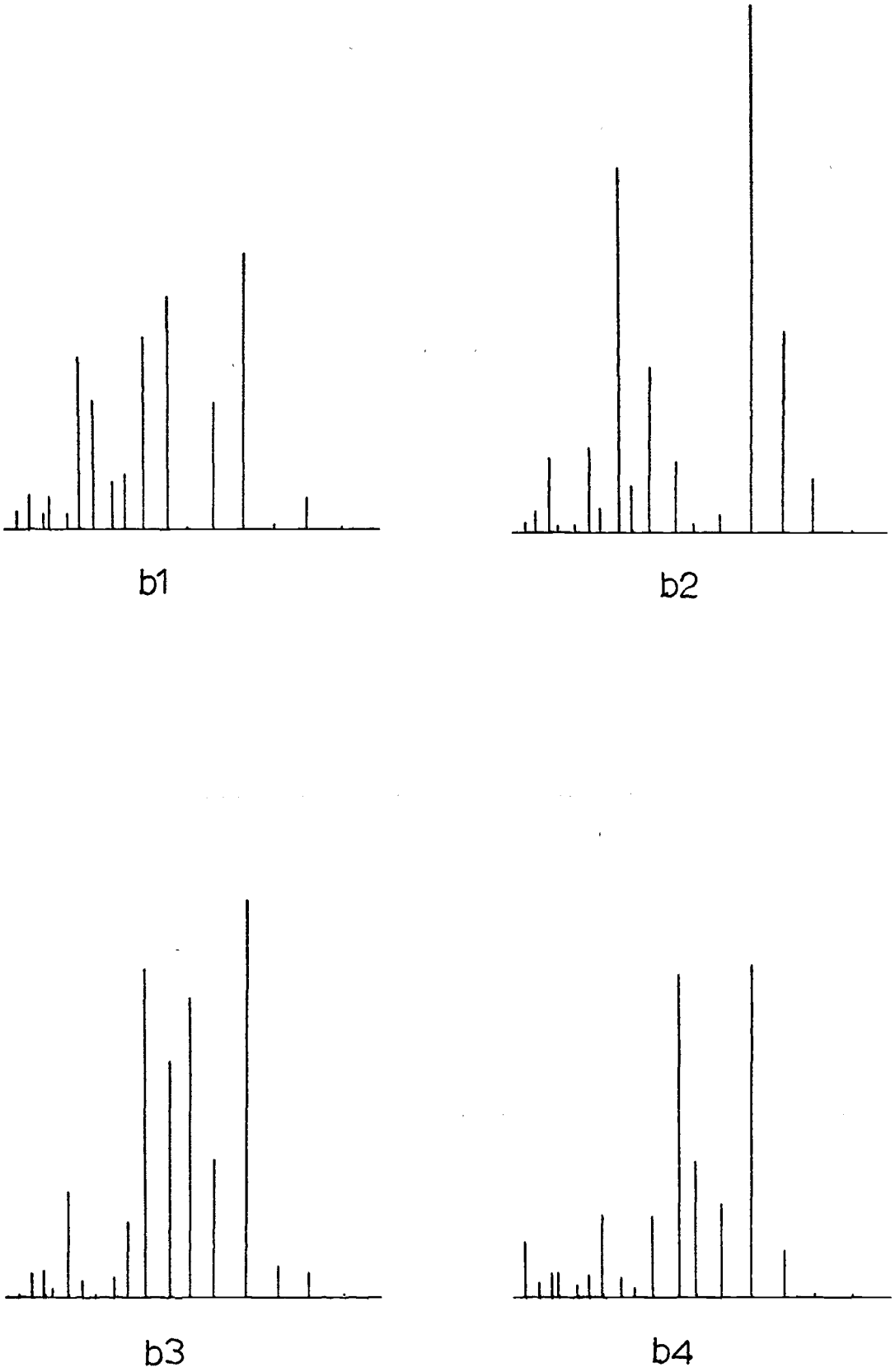
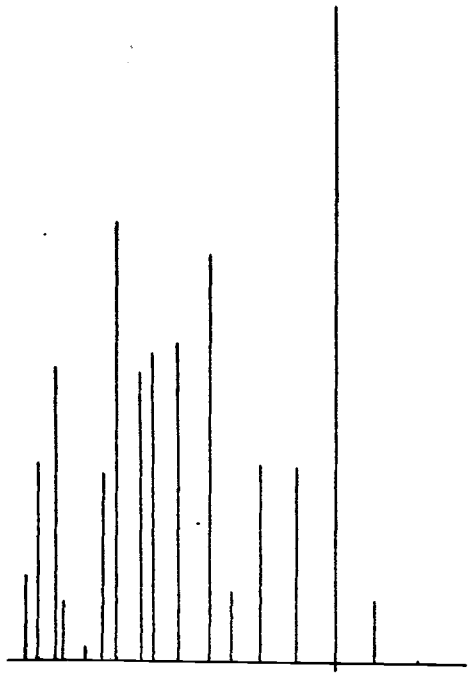
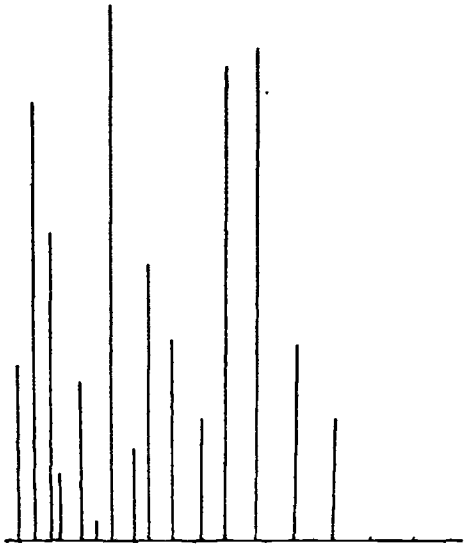


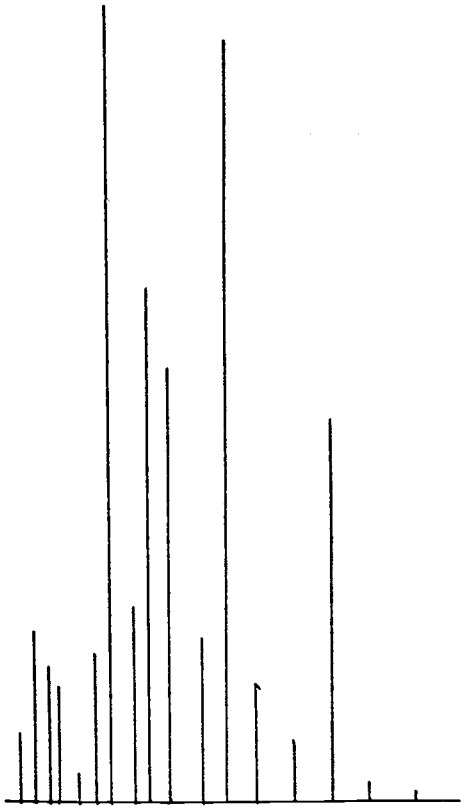

Fig 2



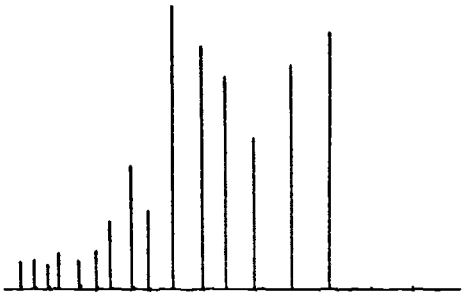
c1



c2

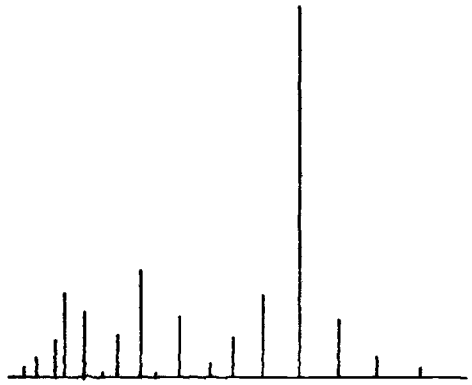


c3

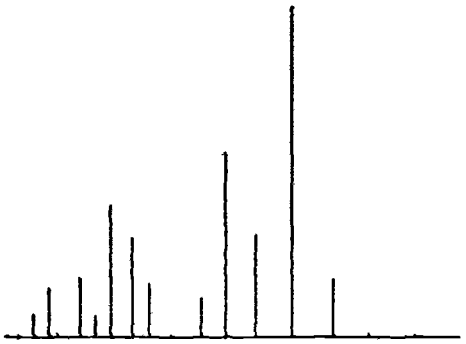


c4

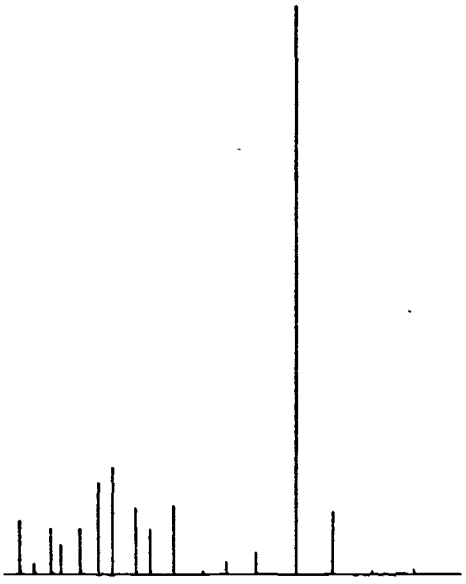
Fig 2



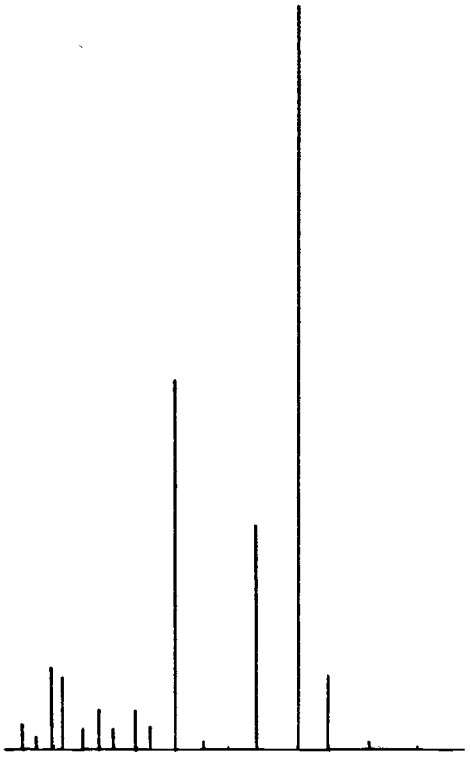
d1



d2



d3



d4

Fig 2

# CHAPTER 9

## PICTURE INPUT AND PROCESSING

The input stage of the general pattern recognition system has so far been ignored to a large extent, perhaps because there are so many apparently obvious solutions. But it is worthy of attention in detail since it is in practice a far from simple task. Without some method of directly, perhaps optically, transferring information contained in an image into a storage medium suitable for subsequent computation, in parallel, great ingenuity is required to solve the problem of transmitting large quantities of data from image to computer.

The most common methods involve some type of scanning procedure, but if this is to be done quickly it can be extremely expensive. The cost of a relatively straightforward system, with the power to carry out selective scanning where required, may be as much as tens of thousands of pounds. Such costly systems are most extensively used in the field of chromosome shape analysis. For more general work, on the non-microscopic level, there are alternatives.

### Cathode ray tube

An opaque photographic slide or similar representation of the data image is placed between a computer-linked cathode ray tube (CRT), of the type used in computer-aided design studies, and a photocell. (See Fig. 1).

a: light detector

b: slide

C: cathode ray tube

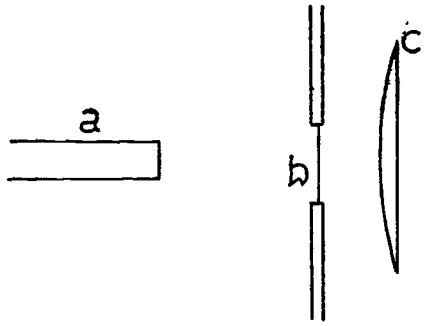



Fig. 1

Each of, say, 1024 x 1024 points on a raster covering 8" x 8" of the cathode ray tube screen is illuminated in turn. Either a two-dimensional array of 1s and 0s (where the light has failed to pass through the opaque silhouette of the shape, or where it has passed through the transparent surround of the slide, and been detected by the photocell), or the co-ordinates of the boundary points of the shape can easily be recorded and stored from such a scan, which takes only a few seconds.

However, apart from the expense of providing a computer with a CRT as a peripheral, there are the following disadvantages and practical difficulties in using the CRT for scanning:

(i) The screen of the CRT has some curvature, and a thick front cover, which introduce geometrical errors and distortions.

(ii) The normal CRT phosphors which are used to illuminate the points on the screen have a long persistence and a high background glow. This means that light from both the background and from the previously illuminated point interfere with the accurate detection of the currently illuminated point.

(iii) The photomultiplier normally provided in the light pen (used as a detector in the same way as a photocell might be used) is not sensitive enough by a factor of about 100.

(iv) An attachment to the CRT to collect light from the screen and concentrate it upon the detector mechanism would prevent simultaneous observation of the screen by the user.

(v) The light sensitivity of the photomultiplier requires that the detector mechanism be near the screen, but it is only at longer range that the geometrical problem - (i) - becomes tolerable.

(vi) The screen is simply too large - it was designed primarily for human observation.

A solution to the above difficulties lies in the use of a slave CRT of much smaller dimensions. The slave CRT would use a short persistence phosphor and would allow simultaneous observation of the main screen. A raster of 2000 x 2000 , or even 3000 x 3000 , could be used on a

slave CRT as small as 2" x 2". In the process of transferring information from the main screen to the slave CRT, an analog to digital converter could be used to give information about grey levels. This would not be possible using only the main screen, except with the addition of a great deal of software.

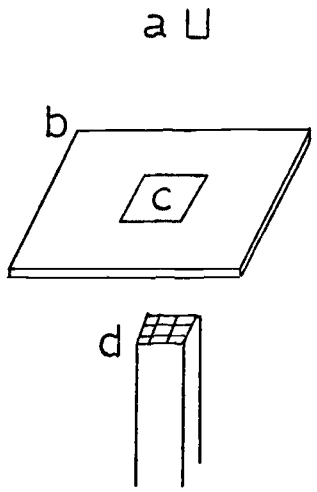
The slave CRT also has the advantages of allowing the measurement of the difference between two images, and can be used as an automatic micro densitometer, with greater ease than the CRT alone. Used as a buffer between the main CRT and the computer store, the slave CRT can input data directly to the store.

Stepping motors - moving stage

Much cheaper, but slower, equipment can be devised using a computer linked mounting for the slide, which can be moved in two orthogonal directions by means of stepping motors to any of 1024 x 1024 different positions. A complete scan with such equipment would certainly be painfully slow, since the speed is affected by the time required after each movement for the vibrations of the motors to die down. However, this time can be utilised for the purpose of carrying out the computations necessary to direct the movements of the stage in accordance with an edge-following algorithm. In this way, the co-ordinates of the boundary points can be recorded. The light source need not be sophisticated, and the detection mechanism consists of a 5 x 5 or 9 x 9 array of photocells,

This edge-following technique could have been used with the CRT, but since a complete scan takes only a few seconds, and the co-ordinates of boundary points can be deduced from it, it is unlikely that the computation involved would be justified.

The system of using a moving stage, stepping motors, and an edge-following algorithm is inexpensive compared with the CRT method.




a: light source

b: moving stage

c: slide

d: light detector

moving stage system

Edge-following

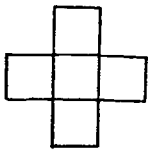


Fig. 2

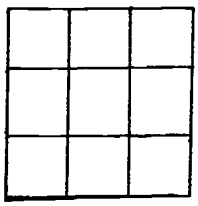


Fig. 3

A cross-shaped array of 5 photocells, as shown in Fig. 2, is sufficient to detect all the boundary points of an image represented by a binary valued data set on a raster of square cells. However, it will also detect some cells which are not boundary cells, and will later reject them. The 3 x 3 array of 9 photocells shown in Fig. 3 will only detect boundary cells.

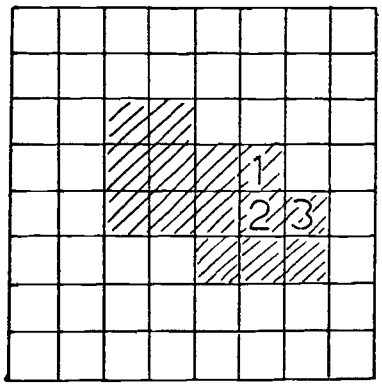



Fig. 4

With refernce to the above Fig. 4, a 5 element array of photocells would move to square 2 from square 1, and then to square 3 . At that stage, square 2 would be eliminated from its list of boundary cells, whilst squares 1 and 2 would remain. With a 9 element array of photocells, the centre cell would move directly from square 1 to square 3 .

Once the co-ordinates of boundary cells have been obtained they can be sorted into left to right, top to bottom, order by a special sorting routine designed for this picture processing work.

One of the advantages of the edge-following algorithm described on pages 85 and 86 is that a 'dual' boundary cell is recorded twice. The boundary cells of Fig. 5 below are 1, 2, 3, 4, 5, 5, 6 . If it is required to reconstruct the original image from the co-ordinates of the boundary cells it is necessary to know that cell 5 is the first internal cell, as well as the last internal cell, as the top line of the shape is scanned.

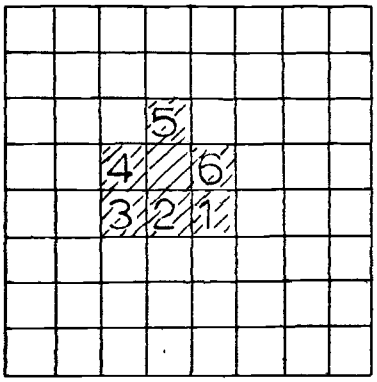



Fig. 5

The edge-following algorithm incorporates the following rules:

(i) A point is a boundary point if one of the four surrounding points is an external point.

(ii) The next boundary point is one of the eight neighbours of the current one.

(iii) The previous boundary point is one of the eight neighbours of the current one.

(iv) In any 3 x 3 array of which the centre point is a boundary point, there are at least three boundary points unless the entire image consists of less than three points.

# CHAPTER 10

# PATTERN GENERATION AND SIMULATION OF GROWTH PROCESSES

## Introduction

Methods of shape analysis were discussed in Chapter 6 and their application to leaf shapes in Chapter 8. They are methods by which the shapes are ultimately defined by a set of numerical values, and suffer from three disadvantages. Firstly, the amount of information to be processed is extremely large for any two-dimensional image. Secondly, the methods are suited to static situations, and cannot easily cope with the development of a shape in time. Thirdly, the difficulty of differentiating between different occurences of the same shape is as yet unresolved.

Particularly in a problem area such as the growth of leaves, and consequent change in leaf shapes with time, there is a more suitable technique for shape analysis which involves less computer time and storage than the alternative of using Walsh functions of one extra dimension to account for time. The problem of inputting large quantities of data is also avoided when this technique simulation of growth processes - is used. The technique consists of a computer program which, when given values for its various parameters, will generate a shape. If one of the parameters is time, then by assigning a series of values to that parameter, a series of shapes, representing the development of the final shape through time, will be generated.

The specification of a shape is that computer program and that set of parameter values which were required to generate it. The methods of Chapter 6 are valuable where the main interest is in statistical work and picture processing; but as a research tool a simulation procedure is of much greater utility. A set of generation rules (the computer program) may be the most efficient means of describing and even classifying two-dimensional shapes; the simulation process may be suggestive of mechanisms of growth in the leaf; hypotheses about the mechanisms of growth can be incorporated into the simulation programs and unambiguously rejected if the patterns generated are at variance with the predicted results of the hypotheses. The field of applications extends beyond leaf shapes to molecule shapes, growth of bacterial moulds, and the development of neuron networks.

The philosophy behind the experiments described in this chapter consists of a belief that parameters should be kept to a minimum until they are well-understood; that parameters need not originally have any obvious biological significance; and that special note should be made of the effect which simple generation rules can have.

Cellular growth

The simplest mathematical model is one in which the shape whose generation is to be simulated is defined on a square array of cells. Those cells which constitute the shape are represented by having the value '1' whilst the remainder have value '0'. This particular represent-

ation is useful if the generated shape is to be stored in a computer, since the storage occupied will be small.

Some work by Eden followed this line of approach (8, 9). He assumed that all cells are identical, immortal (once generated in a simulation, a cell could not disappear), and connected (this followed from the requirement that growth took place only by the appearance of new cells which had to be attached to current boundary cells). He further assumed that there exists a 'time to division' distribution which is the same for all cells.

A cycle of a typical program based on such rules consists of examining each boundary cell in turn; generating a random or pseudo random number, on the basis of which it is decided whether or not the given boundary cell will grow into one or more of its four neighbouring cells, in conjunction with the 'time to division' distribution. If growth is to take place, then further random numbers decide which of the four neighbours is to be grown into, taking into account that some may already be cells of value 1 . The probabilities of growth in the four directions - left, right, up and down - can be varied.

Eden found that results obtained from this simple model fitted quite well one or two biological growth processes. He analysed the results by viewing the process as a branching one, thereby providing an interesting link with the work on vein structures which will be discussed later in this chapter.

The biological process of cell division involves expansion in the size of individual cells and growth at all internal and external points, rather than solely at the periphery. However, Eden found that allowing growth only from the boundary cells and making no allowance for increase in cell size, realistic simulations could still be achieved. The same was found by the present author; and examples of his results are given below.

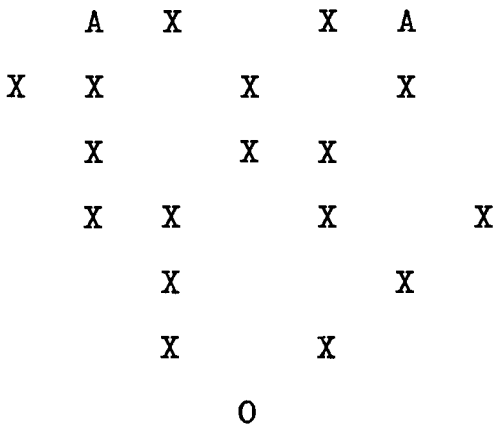
(i) Attractive forces

(a) Growth originates from the cell marked 'O'

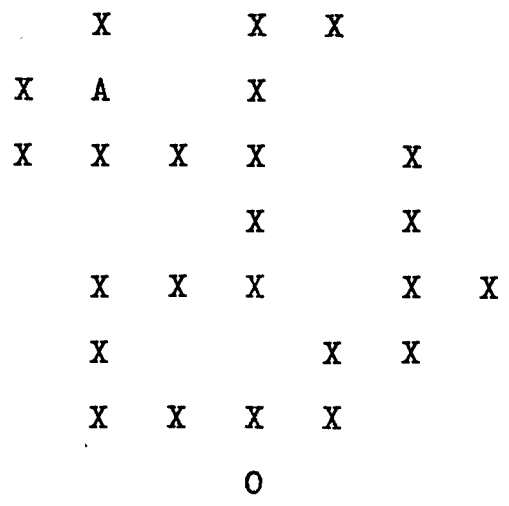
(b) Growth may take place only from boundary cells

(c) Growth may take place only into the four neighbours of a boundary cell, not into the eight with which it shares corners

(d) Growth is attracted towards the cells marked 'A'. These cells may have negative values, which causes growth to be in the direction away from them.




In this example the attractive forces have positive values and there is a low general probability of growth.




In this example, the attractive forces have the same positive values, and growth takes place over the same period of 'time', but the general probability of growth is greater.

Diffusion

The process begins with a concentration of 'material' (represented by the number 100) at the centre of a square cellular array. On each cycle of the process there is a fixed probability of diffusion of material into the neighbouring cells of any cell (external or internal) which does not have the value 0 . The amount of material diffused is proportional to the amount in the cell from which diffusion takes place. In the example below, each cell is assumed to have eight neighbours.

| | | | | | | |
|---|---|---|---|---|---|---|
| stage 0 | | | | 100 | | |
| | | | | 4 | | |
| stage 1 | | | 10 | 69 | 15 | |
| | | | | 2 | | |
| | | | | 2 | | |
| | | | 1 | 12 | 2 | 1 |
| stage 2 | | 3 | 12 | 43 | 10 | |
| | | | 1 | 9 | 1 | 1 |
| | | | | 2 | | |
| | | | | 5 | 2 | |
| | | | 8 | 5 | 7 | 1 |
| stage 3 | 1 | 1 | 11 | 17 | 12 | 4 |
| | | 1 | 3 | 8 | 1 | 4 |
| | | | 1 | 3 | 2 | |
| | | 1 | | 2 | | |

In the above example there was a greater tendency towards horizontal growth than vertical growth, which in turn was more likely than growth in the diagonal directions.

Density restrictions

Growth in a square cellular array is permitted only from boundary cells into their four neighbours. There are equal probabilities of growth in each of the four directions, subject to the conditions that growth tends to be in the directions in which most growth has already taken place, and also into those local areas where the existing density is already highest.

The result of these rules is the development of a shape which is at first unrecognisable, but which gradually resembles an almost perfect diamond shape.

The reasons for this final shape are that growth is only directly in horizontal and vertical directions. In the diagram below, it takes two steps at least for cells 2, 4, 6 and 8 to grow, whereas cells 1, 3, 5 and 7 can grow in one stage.

| | | |
|---|---|---|
| 8 | 1 | 2 |
| 7 | | 3 |
| 6 | 5 | 4 |

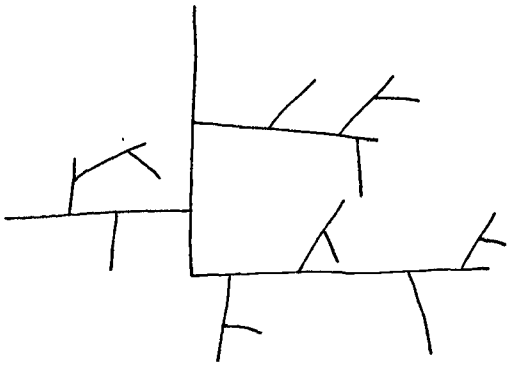
Growth tends therefore to be greatest to the left, right, up and down from the origin; and since growth is more likely in those directions in which most growth has already taken place, the effect is magnified.

Secondly, the four points of the diamond are connected by almost straight lines with smooth profiles because ragged edges cause local areas of high density in between the protrusions. Growth is biased towards areas of high density, so the gaps between protrusions get filled in.

The above are examples of the results which can be obtained using simple rules. By combinations of such rules, realistic effects can be produced.

Two-dimensional branching patterns

Work on two-dimensional pattern generation has been done by Cohen and he mentions leaf shapes in passing (5). His program incorporated six growth rules and five branching rules. The generation process terminated since the extent of growth was in inverse proportion to the quantity of previous growth at any stage. Parameters could be varied during the course of a simulation. The rules were non-probabilistic and did not have obvious biological interpretations.



A typical result of Cohen's program

Grammars and leaf shapes

The mesophyll is that part of the leaf between the veins. Given the vein structure of a leaf the area covered by the mesophyll is easily determined with considerable accuracy. The opposite is not true, since leaves of quite similar shape may have completely different vein structures.

A mathematical model is described here which assumes that the growth and development of a leaf's vein structure can be regarded as a two-dimensional branching process. In fact, the parts of a vein structure may be disconnected at certain stages of its development, and growth may take places at places other than the free ends of the already existing structure.

The notation used to specify a vein structure is provided by a simple generative grammar of only two rules:

$$V \longrightarrow v\,(\,a\,S\,b\,)$$
$$S \longrightarrow V\,(\,S\,)$$

This may be read as 'A vein strcture, V, consists of a vein, v, to which there may or may not be attached a structure S, which is a set of vein structures.' (3).

Fig. 1 , on the next page, shows a vein structure comprising a main vein and three secondary vein structures.

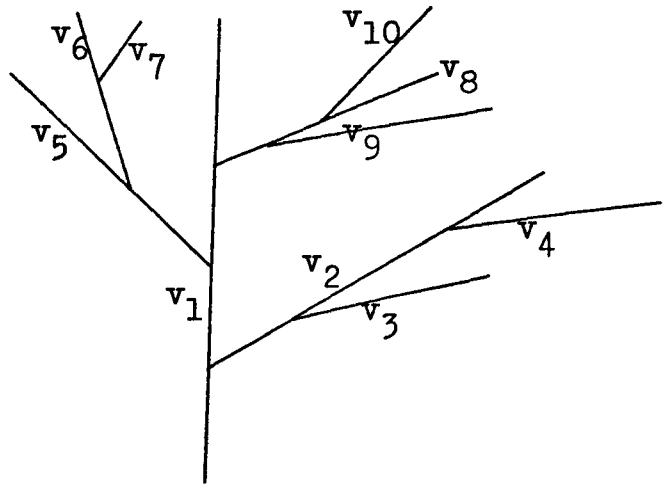


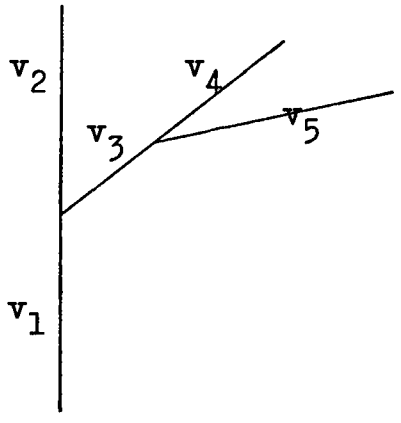

Fig. 1

The sentence - or string of symbols - used to represent Fig. 1 is

$$v_1 \; ( \; v_2 \; ( \; v_3 \; v_4 \; ) \; v_5 \; ( \; v_6 \; ( \; v_7 \; ) \; ) \; v_8 \; ( \; v_9 \; v_{10} \; ) \; )$$

where a and b have been replaced by brackets for easier reading. The numbers used to label the 'v's are only for reference to Fig. 1.

An alternative grammar could be devised under which the representation would be as below, for example:




$$v_1 \; ( \; v_2 \; v_3 \; ( \; v_4 \; v_5 \; ) \; )$$

A 'v' now represents a length of a vein uninterrupted by branching, rather than an entire vein.

In either case, the brackets establish the vein - sub vein relationship. Everything which appears in the brackets after a 'v' is a branch, directly or indirectly, of that v .

A sentence of such a generative grammar may be viewed in a different light altogether, as the result of a stochastic process, i.e. a process in which the current symbol depends only on the previous symbols. A single stage stochastic process is one in which each symbol depends solely upon the previous symbol.

Using the grammar above, a stochastic process can be devised by means of which vein structures can be generated. The grammar can be used to describe vein structures.

A stochastic process of only one stage can be completely described by means of a probability transition matrix. The i,j th element of such a matrix is the probability that the symbol corresponding to row i will be followed by the symbol represented by column j . (In general, that state j will follow state i ).

In the present case, the probability transition matrix is

| | v | ( | ) |
|---|---|---|---|
| v | $p_1$ | $p_2$ | $p_3$ |
| ( | 1 | 0 | 0 |
| ) | $p_4$ | 0 | $p_5$ |

The zeroes indicate that a given symbol cannot be followed by a second given symbol, and the 1 that a certain symbol must always be followed by another specified symbol.

A transition probability matrix, P, is said to be stationary if $P^n$ tends to a limiting matrix, Q, as n tends to infinity. In this case, it does, and

$$Q = \frac{1}{p_2p_4 + p_3 + p_4} \begin{bmatrix} p_4 & p_2 \cdot p_4 & p_3 \\ p_4 & p_2 \cdot p_4 & p_3 \\ p_4 & p_2 \cdot p_4 & p_3 \end{bmatrix}$$

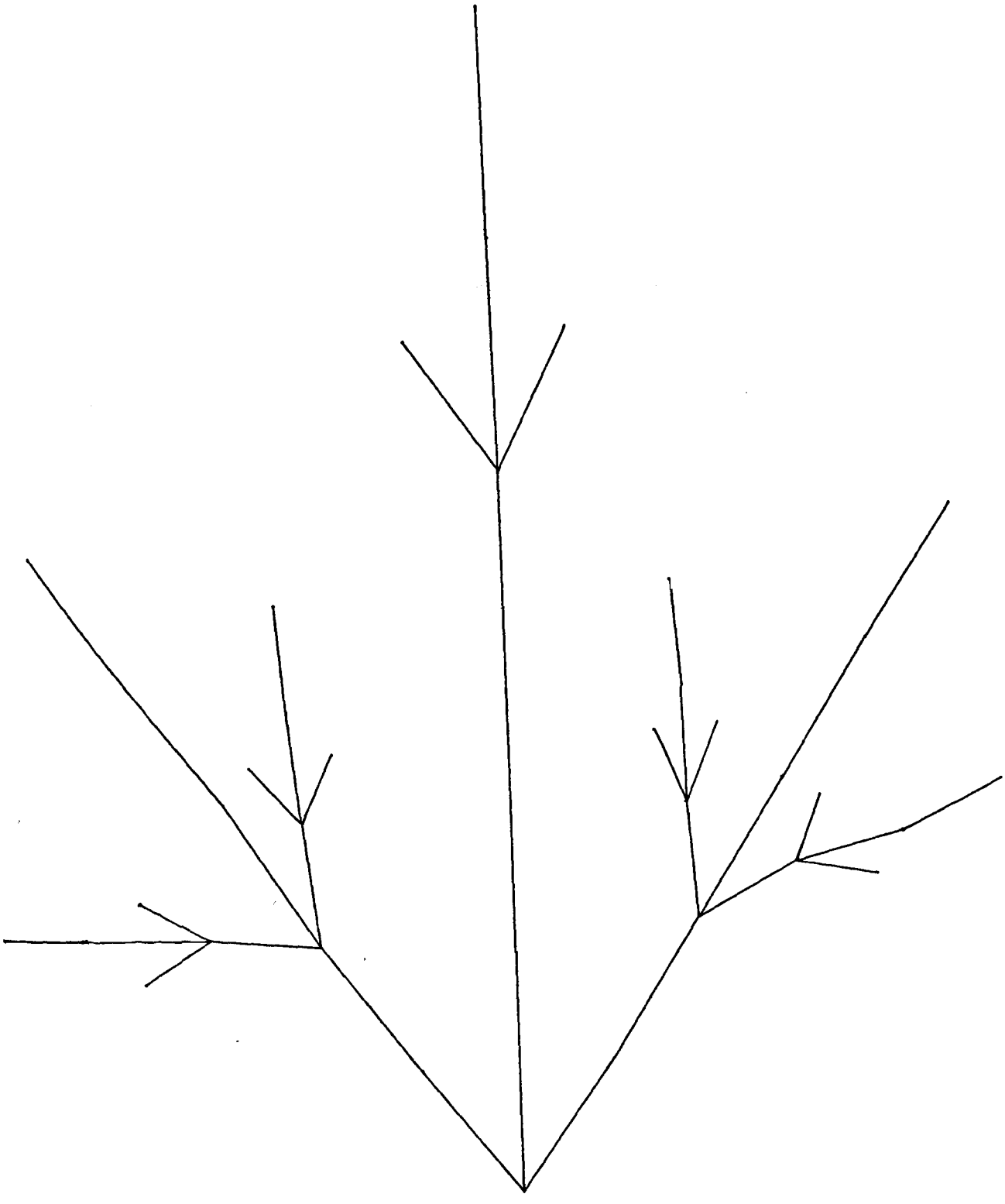
It is a property of such limiting matrices that all rows will be identical. Q tells us the probability of being in a certain state in the long run, i.e. it determines the proportions in which the symbols appear in the long run. Since the frequency of brackets determines the amount of branching in the structure, this can be controlled by choosing a suitable value for $p_2$, $p_3$ and $p_4$.

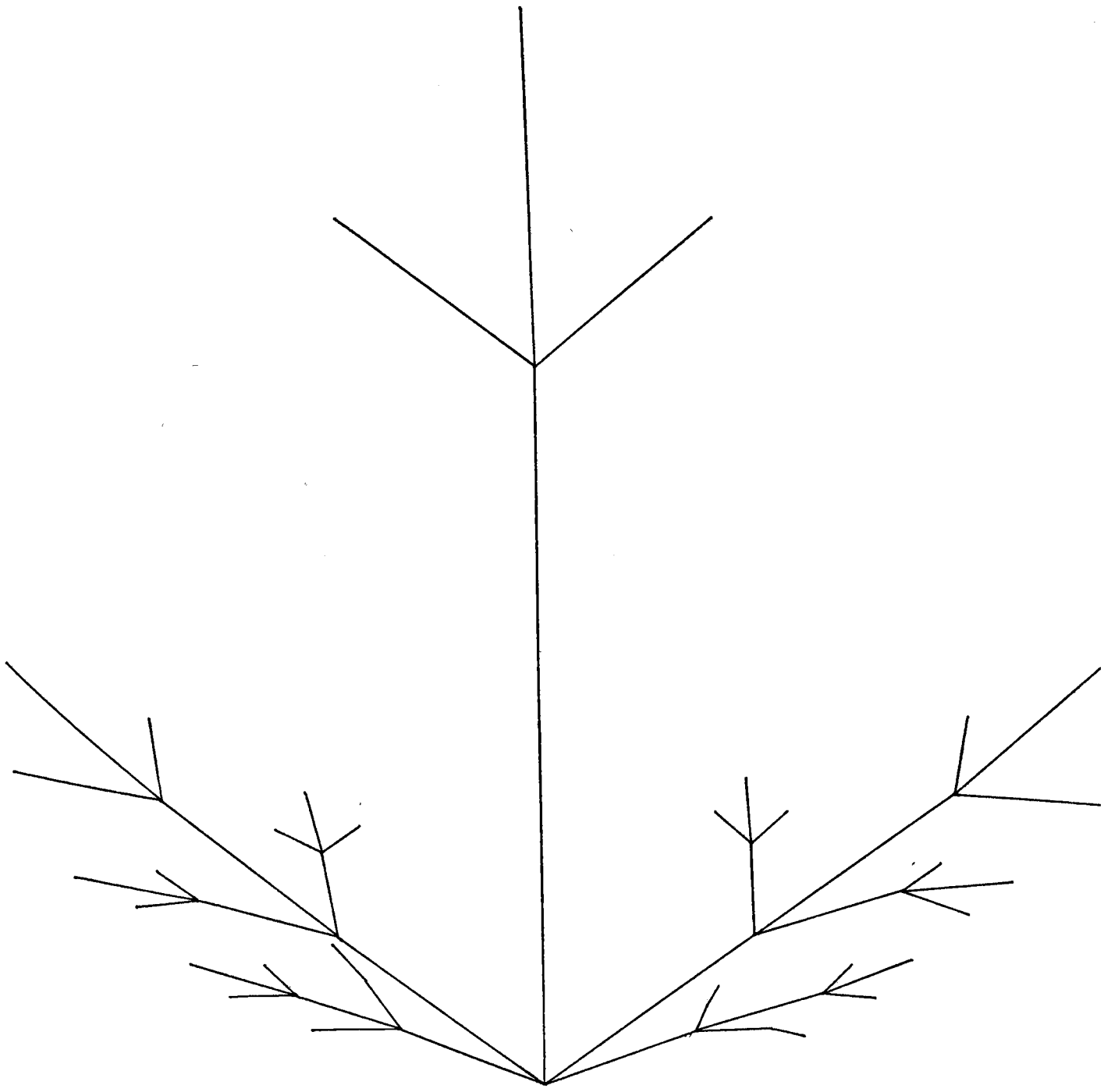
Notice that $p_1$ and $p_5$ do not occur in Q, and that if the process is normally to terminate then the number of opening and closing brackets must be equal, i.e. $p_2 \cdot p_4 = p_3$.

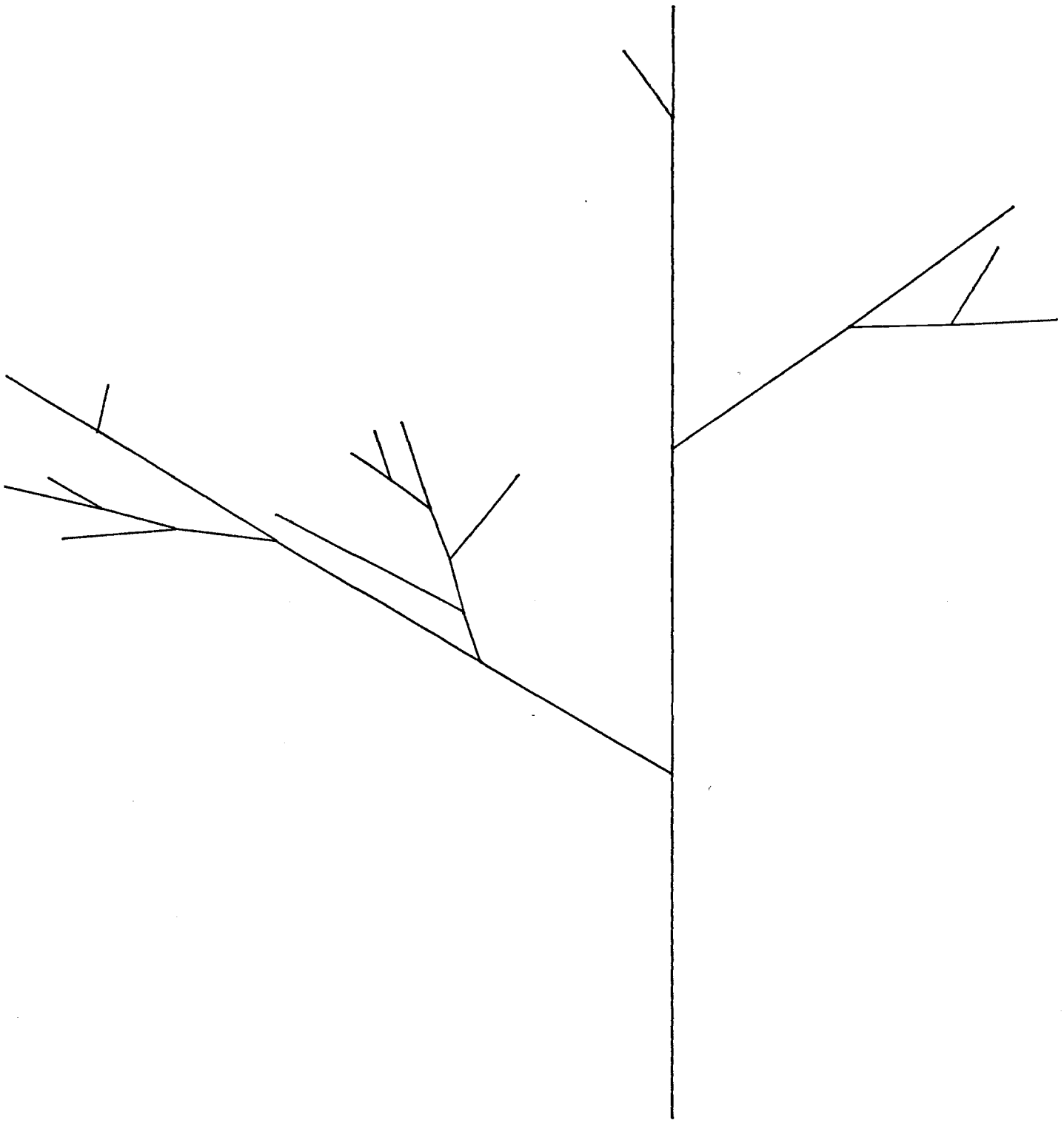
A simulation program was written on the basis of a stochastic process similar to that above, with additional

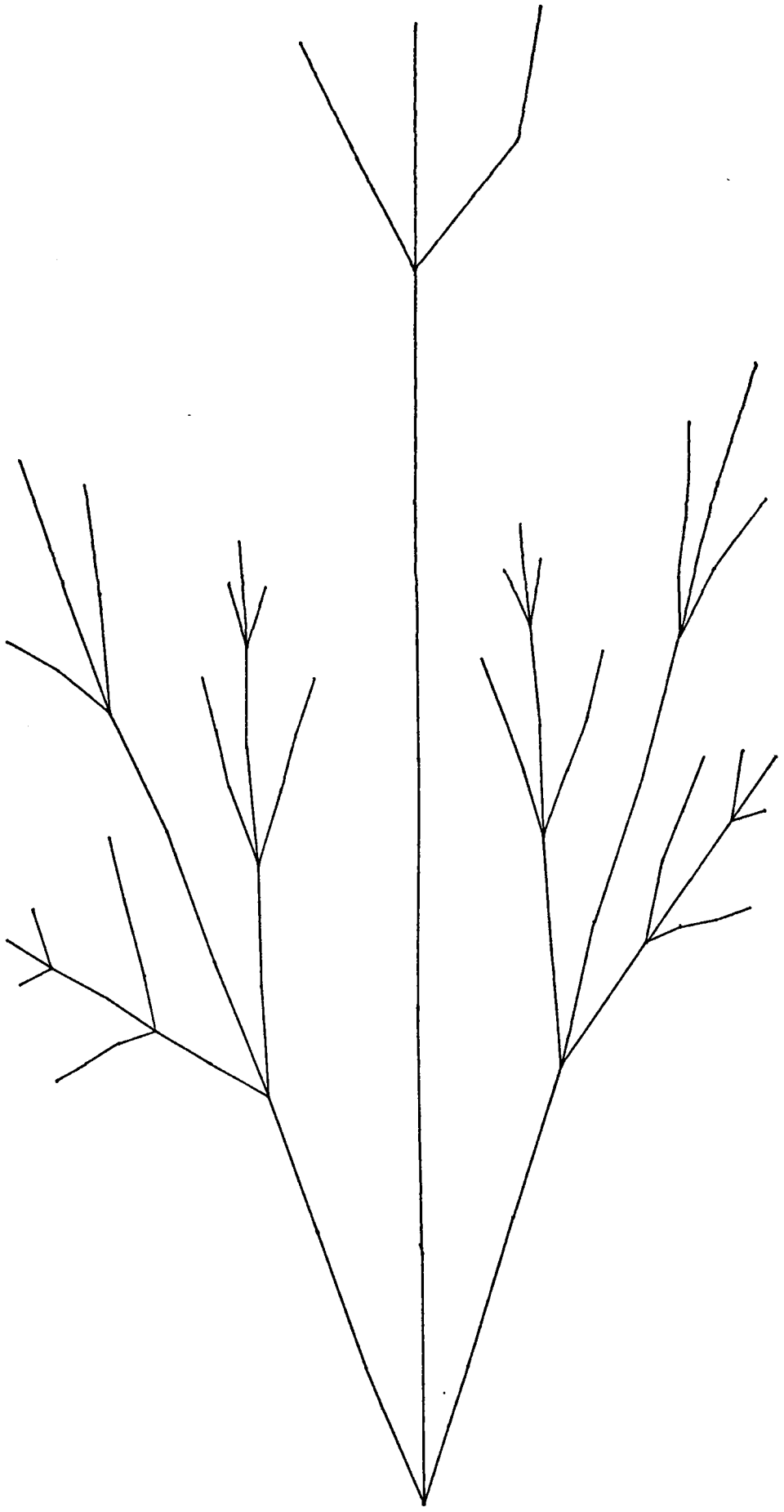
rules to determine the extent of growth and branching, and the angle of branching. The following pages show results of the program.

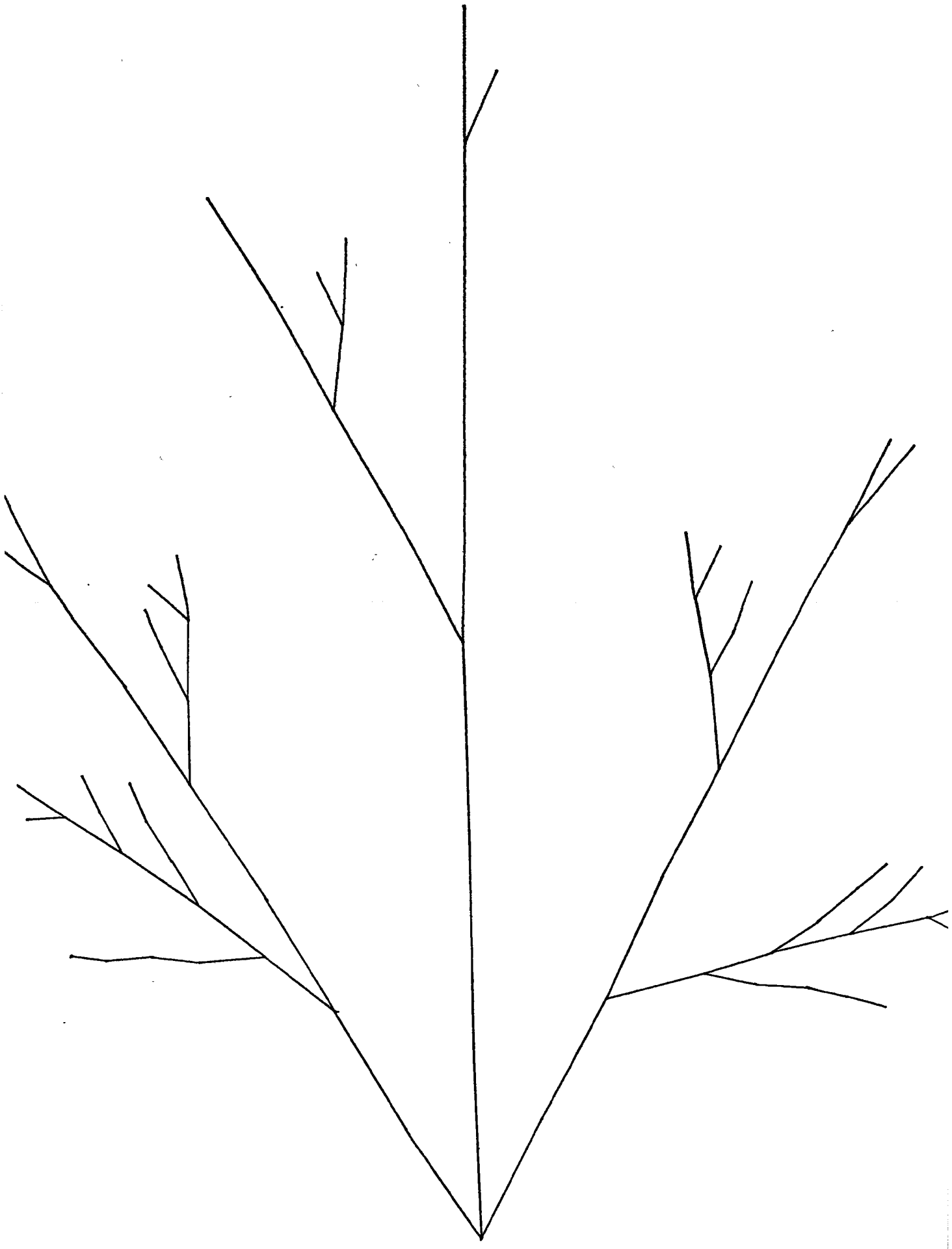
The generation of two-dimensional structures by means of generative grammars is closely connected with the work on pattern description using such grammars by Clowes (4) and others.











# CHAPTER 11

## CONCLUSION

The methods of using Walsh functions which are typified by their use in this thesis as approximators of two-dimensional functions which represent shapes are applicable to a much wider range of problems.

Polyak and Shreider (24) demonstrate the advantages of Walsh functions in mathematical function approximation.

Harmuth (12) examines the role of Walsh functions in communication and signal processing.

'Spatial filtering' of the Walsh transform spectrum has been used in the processing - line removal, deblurring, etc - of photographic images.

Lee's review paper (14) covers applications ranging from the analysis of speech signals to switching theory and logical design.

Everywhere that sets of orthogonal functions are used the Walsh functions, with their binary values and suitability for parallel processing, have advantages over the others from the computational point of view. And where the Walsh transform is applied to binary data (all data in a digital computer is of a binary form) the logical Walsh transform is even more advantageous.

The work of this thesis, particularly the development of the logical transform, is of relevance to problem areas far beyond that of shape analysis.

## REFERENCES

1. Buneman, O. P., "A grammar for the topological analysis of plane figures", Machine Intelligence 5, (eds. B. Meltzer & D. Michie), Edinburgh University Press, 1970.
2. Carl, J. W., "Generalized harmonic analysis for pattern recognition: a biologically derived model", M S Thesis GE/BE/705-1 WPAFB, Ohio Air Force Inst. Tech., 1969.
3. Chomsky, N., "Syntactic structures", Mouton & Co., London, 1957.
4. Clowes, M. B., "A generative picture grammar", Seminar paper no. 6, Computing Research Section, Comm. Sci. & Ind. Res. Org., Canberra, Australia, April 1967.
5. Cohen, D., "Computer simulation of biological pattern generation", Nature, Vol 216, No. 5112, 1967.
6. Coombs, A. W. M., "On the systematic construction of features for automatic character recognition", Proc. IEE NPL conf. on pattern recognition, p 55, 1968.
7. Coombs, A. W. M., "On the construction of an efficient feature space for optical character recognition", Machine Intelligence 4, (eds. B. Meltzer & D. Michie), Edinburgh University Press, 1969
8. Edèn, M., "A probabilistic model for morphogenesis", Symp. on Inf. Th. in Biol., Pergamon Press, New York, 1958.

9. Eden, M., "A two-dimensional growth process", Fourth Berkeley Symp. on Math., Stats., and Prob., Vol. 4, University of California Press, 1961.

10. Gerardin, L. A., & Flament, J., "Geometrical pattern feature extraction by projection on Haar orthonormal basis", Proc. Int. Jt. Conf. on Artificial Intelligence, (eds. D. E. Walker & L. M. Norton), Washington, 1969

11. Harmuth, H. F., "Transmission of information by orthogonal functions", Springer-Verlag, Berlin, Heidelberg, New York, 1969.

12. Harmuth, H. F., "Applications of Walsh functions in communications", IEEE Spectrum, November 1969.

13. Henderson, K. W., "Some notes on the Walsh functions", IEEE Trans. on Elec. Comp., February 1964.

14. Lee, J. D., "Review of recent work on applications of Walsh functions in communications", Proc. Symp. and Workshop on Appl. of Walsh Fns., Washington, 1970.

15. Lu, K. H., "Harmonic analysis of the human face", Biometrics, Vol 21, No. 2, 1969.

16. Meltzer, B , "Speculations on the use of orthonormal functions in the study of morphogenesis", Nature, Vol. 217, No 5124, 1968.

17. Meltzer, B , Searle, N. H., & Brown, R., "Numerical specification of biological form", Nature, Vol 216, No. 5110, 1967.

18. Meltzer, B., & Searle, N. H., "Impotence principle in descriptive morphology", Nature, Vol 217, No. 5135, 1968.

19. Minsky, M., & Papert, S., "Perceptrons", MIT Press, Cambridge, Mass., 1969

20. Narasimhan, R., "A linguistic approach to pattern recognition", Report No. 21, Digital Computer Laboratory, University of Illinois, July 1962.

21. Paley, R. E. A. C., "On orthogonal matrices", J. Math. Phys., Vol 12, pp 311 - 320, 1933.

22. Pease, M. C., "An adaptation of the fast Fourier transform for parallel processing", J. ACM., Vol 15, No. 2, 1968.

23. Pease, M. C., "Organization of large scale Fourier processors", J. ACM., Vol 16, No. 3, 1969.

24. Polyak, B. T., & Shreider, Yu. A., "The application of Walsh functions in approximate calculations", Voprosy.Teor. Matem. Mashin., Coll. II, Moscow, 1962.

25. Searle, N. H., "Shape analysis by use of Walsh functions", Machine Intelligence 5, (eds. B. Meltzer & D. Michie), Edinburgh University Press, 1970.

26. Searle, N. H., "A logical Walsh-Fourier transform", Proc. Symp. and Workshop on Appl. of Walsh fns., Washington, 1970.

27. Sheng, C. L., "Threshold logic", The Ryerson Press/ Academic Press, Toronto/London & New York, 1969.

28. Tallman, O. H., "The classification of visual images by spatial filtering", PhD Thesis, WPAFB, Ohio Air Force Inst. Tech., 1969.

29. Walsh, J. L., "A closed set of normal orthogonal functions", Amer. J. Math., Vol 55, pp 5 - 24, 1923.

(*Reprinted from Nature, Vol. 216, No. 5110, pp. 32–36, October 7, 1967*)

# Numerical Specification of Biological Form

In the study of topics such as morphogenesis, numerical specification of form is particularly important. The authors outline a method, using Walsh functions, which can be used to define a two-dimensional form.

by

B. MELTZER N. H. SEARLE

Metamathematics Unit, University of Edinburgh

and

R. BROWN

Botany Department, University of Edinburgh

THE numerical specification of form is important particularly in the study of morphogenesis. The development of an organ or organism is characterized by a succession of stages each of which may involve distinctive metabolic states and morphological patterns. The change in metabolic state is now being exhaustively examined with considerable success. It is generally recognized that the metabolic state expresses a particular protein pattern. The nature of the pattern and the mechanism involved in determining it can now be examined with some precision. Change in the morphological pattern, however, still eludes precise definition. Simple inspection shows that change in form is as pronounced as change in metabolic state, but while the one can be examined analytically the other cannot. The difference undoubtedly arises from the fact that, while in one connexion certain attributes of the end product can be expressed quantitatively, in the other they cannot. The total quantity of protein and the relative amounts of different types of proteins can be assessed in numerical terms. The particular characteristics of the morphological pattern are not, on the other hand, amenable to extensive numerical treatment. The change in morphological pattern can only be appreciated in terms of a series of diagrams, which do not provide the basis for an analytical characterization of the change or for an examination of the factors by which it is determined.

Moreover, the present limitation imposes another restraint on the analysis of the morphogenetic process, as at present it is difficult to determine the interaction

between the two changing aspects. It cannot be doubted that the metabolic state is linked in some sense to the form by which it is accompanied. Certain metabolic states are likely to promote certain forms of expansion and not others, and localization of metabolic states is likely to lead to differential growth rates which will promote the development of a particular form. The work on the genetical determination of readily distinguishable forms (in, for example, tomato leaves[1]) indicates as much. At the same time it is not inconceivable that the change in metabolic state may itself be affected by the change in spatial configuration.

Clearly an elaboration of the techniques available for defining biological form is desirable, and the present paper is intended as a contribution to this problem. The situation is simplified when the pattern is two-dimensional and attention has therefore been restricted to foliage leaves; an analysis of change in form with time, however, has not been attempted. The primary purpose has been to elaborate a technique which defines certain quantitative characteristics of the mature form. Several investigations have been concerned with the factors involved in determining leaf shape in particular species, or at different levels along the shoot of a single plant. Melville[2] described a technique for analysing shape in leaves of species of *Ulmus* and several workers have examined the change in shape along a shoot by measuring the lengths of particular axes (Njoku[3]). These techniques are not, however, generally applicable, and the purpose here has been to elaborate a technique which is applicable to all leaves and all stages of development.

In the method described here the two-dimensional pattern is represented by a function taking the values 1 and 0. This function is analysed into a sum of weighted basic pattern functions. The process is very similar to spectral analysis, and in fact the sinusoidal functions of the latter and our basic functions are particular examples

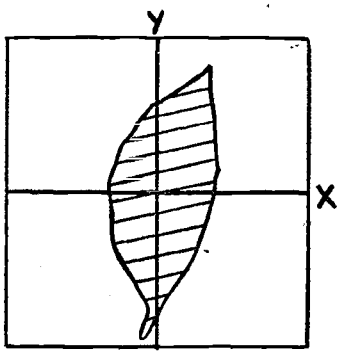



Fig. 1. Object in frame as function.

of the very useful class of orthonormal functions. The method can be straightforwardly generalized to 3-dimensional forms. An initial application of the method to six leaves of *Althea* species and two of *Coleus* species is described.

Consider a two-dimensional form like a leaf, enclosed in a square frame with position specified, say, by Cartesian rectangular co-ordinates, $x$ and $y$. If we assign the value 1 to points where the object is and 0 elsewhere, we shall have defined a function of position $F(x,y)$, taking only 1 or 0 as possible values, which completely fixes the object in the frame. If as in Fig. 1 we shade those portions of the frame where $F(x,y)$ has the value 1, we have a vivid representation not only of the object but of this function also. This function can be expressed as a weighted sum of certain "basic" functions which we denote by means of two indices, as follows

$$f_{0,0}(x,y),\ f_{0,1}(x,y),\ f_{1,0}(x,y),\ f_{0,2}(x,y),\ f_{1,1}(x,y),\ \ldots,\ f_{i,j}(x,y),\ \ldots \text{ etc.},$$

which take only the values 1 and $-1$. Fig. 2 gives a representation of a selection of these basic functions with the regions of value 1 shaded in. If

$$c_{0,0},\ c_{0,1},\ c_{1,0},\ \ldots\ c_{i,j}\ \ldots,\ \text{etc.},$$

are the respective weights of the basic functions required for representing the object, then the object function is given by

$$F(x,y) = c_{0,0}\, f_{0,0}(x,y) + c_{0,1}\, f_{0,1}(x,y) + \ldots\ c_{i,j}\, f_{i,j}(x,y) + \ldots.$$

The weights $c_{i,j}$ can be calculated in a rather simple manner and have a simple intuitive interpretation: $c_{0,0}$ equals the area $A$ of the form, while every other

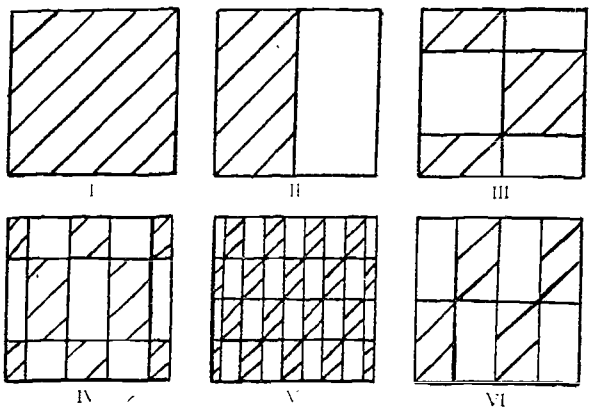



Fig. 2. Selection of basic Walsh patterns; I, $f_{0,0}$; II, $f_{1,0}$; III, $f_{1,2}$; IV, $f_{4,2}$; V, $f_{8,2}$; VI, $f_{1,3}$.

weight is a measure of the correlation between the object pattern and the corresponding basic pattern; it is in fact equal to one-half the correlation coefficient, positive values indicating positive and negative values negative correlation.

The weights tend to get smaller as the indices $i$ and $j$ of the basic function increase. This will always be the case and is in fact a necessary condition for the success of the method, for the set of basic functions is actually infinite in number and one requires that the sum converges to a suitable degree of approximation to its correct value in a finite number of terms. The closeness of the approximation can be easily judged, for if one includes only a finite number of the weights, it can be shown that the pattern reconstituted from the corresponding basic patterns bears a correlation to the true pattern given by the ratio of the sums of the squares of the weights to the area $A$. Thus if all the weights are used, the sum of their squares should be precisely equal to $A$. So the defined ratio can be used to give the percentage accuracy of the represenation.

Qualitative features of the forms are reflected in the values of the weights. For example, patterns which are symmetrical about one of the axes must have certain weights zero, and small-scale features in the form reveal themselves in correspondingly large values of some of the higher-order weights. It is clear that it would not be difficult to devise numerical measures of qualitative features of the pattern such as degree of symmetry or degree of fine-structure, in terms of the squares of the weights.

The weights calculated do not as they stand specify the form. This can be seen immediately, for if one merely shifted, or rotated, or uniformly contracted or expanded the leaf pattern of Fig. 1, the resulting correlation with the basic function patterns would obviously alter, and so the calculated weights would alter. There are two ways in which one may deal with this problem.

The first method, while involving a considerable computational load, provides an exact specification of form with no arbitrary features. Suppose we fixed on one of the basic functions, say, $f_{i,j}$, and calculated the values of $2c_{i,j}$ for all possible positions, orientations and changes of size of the leaf form within the frame, one would obtain a set of values lying in the range $-1$ to $+1$. The algebraically largest of these, that is, the most positive one, would be a measure of the resemblance of the leaf to this basic $f_{i,j}$ pattern, for it gives the maximum correlation possible when one has abstracted the irrelevant factors of location, orientation and size. For a given form. and a given set of basic functions, these maximal weights are uniquely determined, thus constituting an appropriate set of numbers for specifying the form.

In the second method, for a given series of forms—for example, different stages in the development of a leaf—one fixes on some standardization of size, location and orientation in the frame. For a series of the kind mentioned, if one chooses the standardization skilfully, the results can be meaningful and useful. If, however, one wishes to compare one series of forms with another considerably different, it becomes much more difficult to choose suitable standardization for both sets, which would make comparisons meaningful. For the weights calculated are not determined only by the shape of the objects concerned, but also by the particular choice of standard size, position and orientation. However skilful this choice, there is always an element of arbitrariness about it.

There is little doubt that the maximal-weight method is an unequivocal and better method than one using standardization. Because of the latter's comparative simplicity in respect of computation, however, it was used in the initial work reported.

Some experience has now been obtained in analysing leaf patterns. Typically, a coarse analysis giving only the first 16 weights might yield 35 per cent accuracy, while finer ones might give 70 per cent for 64 and 90 per cent for 256 weights. One may therefore have to deal with as many as 256 weights or more.

For purposes of initial and overall survey, it is useful to condense the information contained in the weights. This can be done by grouping weights corresponding to a given structural resolution as follows: with each basic function, $f_{i,j}$, one associates a "resolution", defined as the average distance between successive discontinuities, in both the $x$ and $y$ directions, in the corresponding pattern. If the side of the frame is 1, then

$$\text{Resolution} = \frac{2}{i+j}$$

One adds the sums of the squares of the weights corresponding to a given resolution, and plots these against resolution grouped over equal intervals as in Fig. 3.

These histograms give an intuitively satisfying presentation of the analysis. The large-resolution end of the histogram gives the coarse-structure, "area" type of information about the form; the small-resolution end gives information about fine-structure. The analogy of these histograms to graphs of optical spectra is clear. The abscissa chosen is analogous to wavelength in optics. If the reciprocal of resolution had been used for the abscissa, the parameter would have been analogous to frequency; this, however, appeared to be rather unsuitable, for it represents the fine-structure components as being orders of magnitude smaller than the coarse-structure

ones. This is inappropriate, for it is the fine-structure components which really characterize the shape.

Grouping also has an incidental advantage in that it tends to reduce the effects of the arbitrariness introduced by standardization.

## Analysis of Leaves

The preliminary results presented refer to six successive leaves taken from a shoot of *Althea* species and two leaves taken from the same shoot of a plant of *Coleus* species. The outlines of the leaves were traced and to simplify computation in the initial stages were symmetrized. With the *Coleus* leaves the apex was displaced to bring it into the same line as the axis of the leaf. With the *Althea* leaves the right side was made a mirror image of the left. In the *Coleus* leaf the serrations at the edge were disregarded and the margin was taken to be a line drawn through the serrations. In all cases the form was taken to be that defined by the lamina and the outline that was analysed was drawn through the point of attachment of the petiole to the lamina.

The results are given in Table 1 and plotted in Fig. 3 in histogram form. Each histogram is accompanied by a tracing of the leaf from which it is derived. The histograms give only the "low resolution" weights in the range 0–0·667. The large resolution weights are less significant because they are heavily affected by the particular standardization chosen. The six *Althea* leaves (I–VI, Fig. 3), although they are taken from successive nodes, show a progressive change from a form which is almost entire to one that is sharply palmate. This is reflected in the tendency in the histograms of certain fine structure components to become more prominent. The two *Coleus* leaves (VII and VIII, Fig. 3) although they are also entire are nevertheless lanceolate. Their histo-

Table 1. THE RESULTS FROM WHICH FIG. 3 WAS PLOTTED

Nos. I to VI are leaves from the species *Althea*, and Nos. VII and VIII are from the *Coleus* species.

| Resolution | (Weights)$^2 \times 10^5$ *Althea* | | | | | | *Coleus* | |
|---|---|---|---|---|---|---|---|---|
| | I | II | III | IV | V | VI | VII | VIII |
| 0–0·067 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0·067–0·133 | 876 | 933 | 1,216 | 1,057 | 1,288 | 963 | 702 | 865 |
| 0·133–0 2 | 1,231 | 1,597 | 2,102 | 1,777 | 2,998 | 3,497 | 1,933 | 1,695 |
| 0·2–0·267 | 1,581 | 1,248 | 976 | 1,377 | 791 | 510 | 599 | 801 |
| 0·267–0·333 | 3,200 | 2,471 | 3,515 | 3,116 | 5,083 | 5,798 | 1,515 | 1,494 |
| 0·333–0·4 | 2,238 | 2,375 | 1,917 | 2,539 | 2,335 | 2,004 | 779 | 978 |
| 0·4–0·467 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0·467–0 533 | 1,600 | 2,036 | 1,489 | 795 | 669 | 480 | 1,052 | 754 |
| 0·533–0·6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0·6–0·667 | 39 | 63 | 232 | 222 | 79 | 227 | 697 | 1,328 |
| 0·933–1 | 8,496 | 7,940 | 5,885 | 3,727 | 3,710 | 4,436 | 11,711 | 10,104 |
| 1·933–2 | 1,893 | 2,347 | 3,192 | 5,464 | 2,733 | 1,036 | 1,487 | 2,638 |

grams are similar to each other but in type are remarkably different from those of the first two leaves of the *Althea* series. It may be noted that the change in the form of the *Althea* leaves is a reflexion of a corresponding change in the morphogenetic activity of the terminal meristem from which they are derived. The progressive change in the histogram pattern is therefore a numerical characterization of the change in the morphogenetic activity of the meristem.

## Mathematical Basis

The basic pattern functions $f_{i,j}$, which are functions of two variables $x$ and $y$, are synthesized from certain functions of a single variable which were introduced into mathematical analysis by Walsh[4]. Walsh's set of functions constitute what is known as a complete orthonormal set; it is this property which they share with the sine and cosine functions of Fourier analysis, which makes them suitable for our purposes.

Each of the basic pattern functions $f_{i,j}(x,y)$ is defined as

$$f_{i,j}(x,y) = f_i(x)\, f_j(y)$$

where $f_i$ and $f_j$ are the Walsh functions of index $i$ and $j$ respectively. $f_i(x)$ is defined as follows:

Let $i = d_1 d_2 \ldots d_r$ be the binary representation of the index $i$, where the $d$'s are 0 or 1, but $d_1$ is 1. Let $s$ be the value of 1 plus the binary number $d_2 d_3 \ldots d_r$. Then

$$f_i(x) = \varphi_r^s(x)$$

where $\varphi_r^s(x)$ is defined recursively as follows. Two integral indices $n$ and $k$ are used in the definitions

$$\varphi_0(x): \quad \text{For } -\tfrac{1}{2} \leqslant x < \tfrac{1}{2} \qquad 1$$

$$\varphi_1(x): \quad \text{For } -\tfrac{1}{2} \leqslant x < 0 \qquad 1$$

$$0 \leqslant x < \tfrac{1}{2} \qquad -1$$

$$\varphi_2^1(x): \quad \text{For } -\tfrac{1}{2} \leqslant x < 0 \qquad \varphi_1(2x+\tfrac{1}{2})$$

$$0 \leqslant x < \tfrac{1}{2} \qquad -\varphi_1(2x-\tfrac{1}{2})$$

$$\varphi_2^2(x): \quad \text{For } -\tfrac{1}{2} \leqslant x < 0 \qquad \varphi_1(2x+\tfrac{1}{2})$$

$$0 \leqslant x < \tfrac{1}{2} \qquad \varphi_1(2x-\tfrac{1}{2})$$

For all $n > 1$ and values of $k$, 1, 2, 3 . . . $2^{n-1}$:

$$(2x\varphi_{n+1}^{2k-1}(x): \quad \text{For } -\tfrac{1}{2} \leqslant x < 0 \qquad \varphi_n^k(2x+\tfrac{1}{2})$$

$$0 \leqslant x < \tfrac{1}{2} \qquad (-1)^{k+1}\varphi_n^k(2x-\tfrac{1}{2})$$

$$\varphi_{n+1}^{2k}(x): \quad \text{For } -\tfrac{1}{2} \leqslant x < 0 \qquad \varphi_n^k(2x+\tfrac{1}{2})$$

$$0 \leqslant x < \tfrac{1}{2} \qquad (-1)^k\varphi_n^k(2x-\tfrac{1}{2})$$

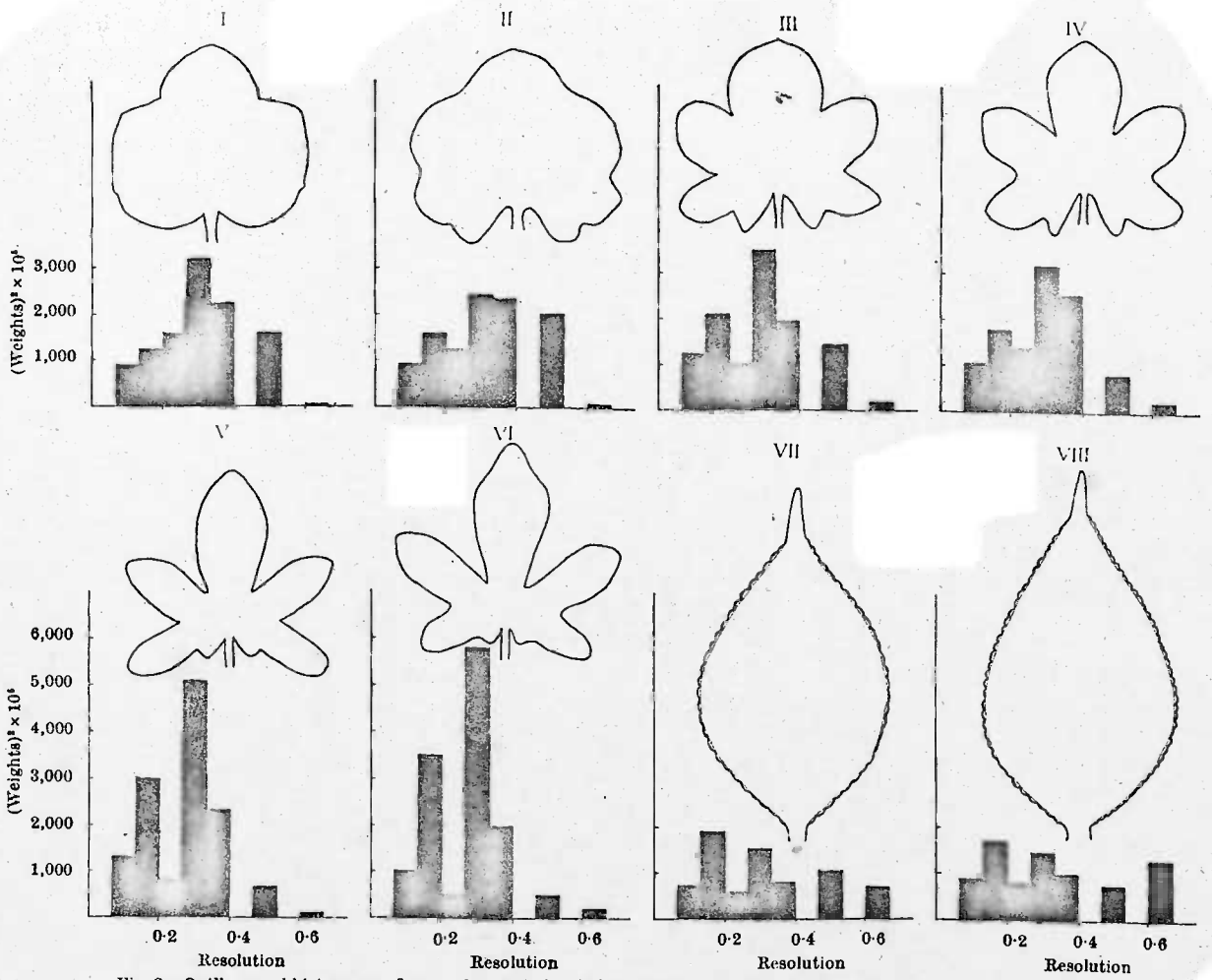



Fig. 3. Outlines and histograms of squared weights for six leaves of species of *Althea* (I–VI) and for two leaves of species of *Coleus* (VII and VIII).

This definition appears complicated, but the construction of the functions is actually quite simple. The functions are defined for values of $x$ between $-\frac{1}{2}$ and $+\frac{1}{2}$. The first function $\varphi_0$ has the value 1 over this domain. The next function $\varphi_1$ has the value 1 over the first half of the domain and the value $-1$ over the second half. The next functions squeeze $\varphi_1$ into the first half of the domain and, respectively, $\varphi_1$ and $-\varphi_1$ into the second half of the domain. The next four functions repeat the same process on each of the previous two. The following sixteen functions repeat the same process on each of the previous four, and so on, and so on.

The orthonormal properties of the functions are expressed by the following conditions

$$\text{Orthogonality: } \int f_i(x) f_j(x) \mathrm{d}x = 0 \text{ for } i \neq j$$

$$\text{Normality: } \int f_i^2(x) \mathrm{d}x = 1 \text{ for all } i,$$

where the integral is over the domain $-\frac{1}{2}$ to $+\frac{1}{2}$ of $x$.

Our indexing of these Walsh functions is a very convenient one, for it turns out that $fi$ has precisely $i$ discontinuities, that is, jumps between $+1$ and $-1$, and that the parity of the function is equal to the parity of the index $i$.

It follows from the orthonormality of the one-dimensional Walsh functions that the basic pattern functions

$$f_{i,j}(x,y) = f_i(x) f_j(y)$$

are also orthonormal, with the integral concerned now being a double integral over a square of side equal to 1. From the orthonormality it follows that the weights $c_{i,j}$ are given by

$$c_{i,j} = \int\int F(x,y)\, f_{i,j}(x,y)\, \mathrm{d}x\, \mathrm{d}y$$

To find how this weight is related to the correlation between the object pattern and the basic function pattern, we set

$$g(x,y) = 2F(x,y) - 1$$

thereby ensuring that $g(x,y)$ represents the object pattern by 1's and $-1$'s, unlike $f(x,y)$ which represents it by 1's and 0's. Hence

$$\begin{aligned} c_{i,j} &= \int\int [\tfrac{1}{2} + \tfrac{1}{2} g(x,y)]\, f_{i,j}(x,y)\, \mathrm{d}x\, \mathrm{d}y \\ &= \tfrac{1}{2} \int\int g(x,y)\, f_{i,j}(x,y)\, \mathrm{d}x\, \mathrm{d}y \end{aligned}$$

provided $i$ and $j$ are not both 0. It is also easy to see that $c_{0,0}$ equals the area of the form represented by $F(x,y)$.

Note that it also follows from the orthonormality that

$$A = \int\int [F(x,y)]^2 \mathrm{d}x\mathrm{d}y = c_{0,0}^2 + c_{0,1}^2 + c_{1,0}^2 + \ldots + c_{i,j}^2 + \ldots$$

where $A$ is the area; hence $c_{i,j}$ tends to 0 as the indices $i$ and $j$ tend to infinity.

It is easily seen, from the parity properties of the Walsh functions, that for a form symmetrical about the $y$-axis the weights $c_{i,j}$ are zero for all odd $i$, and for one symmetrical about the $x$-axis they are zero for all odd $j$. From the oscillation properties one also sees that small-scale features of a form will be indicated by the $c_{i,j}$ with high indices—the latter obviously will not be much affected by large scale features.

Received August 23, 1967.

[1] Srb, A. M., in *General Genetics* (W. H. Freeman, London, 1965).
[2] Melville, R., *Ann. Bot., N.S.*, 1, 673 (1937).
[3] Njoku, E., *New Phyl.*, 55, 91 (1956).
[4] Walsh, J. L., *Amer. J. Maths.*, 45, 5 (1923).

Printed in Great Britain by Fisher, Knight & Co., Ltd., St. Albans.

(*Reprinted from Nature, Vol.* 217, *No.* 5135, *pp.* 1289–1290, *March* 30, 1968)

# Impotence Principle in Descriptive Morphology

It was recently proposed[1] that biological form should be specified by an expansion of the characteristic function of the form as a weighted sum of the elements of a complete set of orthonormal functions. The method was applied to two-dimensional forms, using Walsh functions[2]. In the discussion it was suggested that each weight might be maximized with respect to all possible transformations of the form caused by any combination of displacements, rotations and uniform contractions or expansions. It was conjectured that this sequence of "maximal" weights constituted an unequivocal specification of the form, that is to say, two different forms would always be represented by different sequences. In the case of the Walsh functions, these weights would be real numbers lying in the interval $-\frac{1}{2}$ to $+\frac{1}{2}$, except for the first one which would lie in the interval 0 to 1.

If one accepts the notion of form in complete generality, this conjecture is false. We have not been able to construct an actual non-trivial counter-example in the case of two-dimensional forms in a square frame that we had discussed[1]; but we have found one for a similar system using a fixed circular frame, in which the form is blown up or contracted so as to just fit into the frame and only rotational transformations need to be considered. This counter-example led us to think that there may be some inherent limitation on the possibilities of mathematical description in morphology, and this is indeed the case. We prove that it is impossible uniquely to specify all three-dimensional or two-dimensional forms by finite or denumerable sequences of real numbers.

We prove this for two-dimensional forms—it will then hold for three-dimensional forms, because all two-dimensional ones can obviously be put in one-to-one correspondence with a subset of all three-dimensional forms.

For definiteness, consider a Cartesian co-ordinate system in the plane and let $f(x.y)$ be the characteristic function representing one specimen of a form. that is, it takes the value 1 at points covered by the specimen and 0 elsewhere.

Let $f\alpha(x,y)$ be the characteristic function representing any other specimen of this form, that is, the original form transformed by any combination of displacements, rotations and finite uniform contractions or expansions,

Let $S_f = \{f\alpha(x,y)\}$ be the set of all such transformed functions derived from $f(x,y)$. The cardinality of $S_f$ is clearly $c$, the cardinal number of the continuum.

The cardinality of the set of all characteristic functions $f(x,y)$ whatever is obviously $2^c$. Because the $S_f$s constitute a partition of the set of all these characteristic functions into disjoint subsets, it follows that if $k$ is the cardinal number of the set of $S_f$s, then

$$k \times c = 2^c \tag{1}$$

$k$ is clearly the cardinal number of all possible two-dimensional forms, each $S_f$ corresponding uniquely to a single form.

Suppose it were possible to assign to each $S_f$ a finite or denumerable sequence of real numbers, expressed for example in binary digits. To each such sequence we can make to correspond in one-to-one fashion a single real number, for example, if $d_{ij}$ is the $j$th digit of the $i$th real number in the sequence one could form the real number $d_{11}\ d_{12}\ d_{21}\ d_{13}\ d_{22}\ d_{31}\ \ldots$ and so on, where in the obvious systematic way one writes down successively the digits, the sum of whose index numbers is respectively 2, 3, 4, . . ., and so on. Hence one would have been able to set up a one-to-one matching correspondence between the $S_f$s and the real numbers or a subset of them. One must therefore have

$$k \leq c \tag{2}$$

From (1) and (2) it would follow that

$$2^c \leq c \times c$$

But

$$c \times c = c$$

Hence

$$2^c \leq c, \text{ which is impossible.}$$

Thus we see that there just are not enough real numbers for the unique specification of all possible forms.

This result does not mean that one should give up the numerical approach to morphological description. For, it depends on interpreting form in its fullest possible generality—for example, a leaf with an infinite number of smaller and smaller holes would under this rubric be a form. Clearly, natural forms would constitute some subset of all forms, and it would be profitable to look for some general model for characterizing them. For such a restricted set of forms it may well turn out that real numbers or perhaps even rational numbers would be sufficient for unequivocal description. One possibility, suggested by Professor J. A. Robinson, is—in the two-dimensional case, for example—to consider only forms whose boundary can be approximated to any desired

degree of accuracy by a finite number of polygons other possibilities readily spring to mind

BERNARD MELTZER
NIGEL SEARLE

Metamathematics Unit
University of Edinburgh

Received February 26 1968

[1] Meltzer B Searle N H and Brown I *Nature* **216** 32 (1967)
[2] Walsh J L *Amer J Math* **45** 5 (1923)

Printed in Great Britain by Fisher Knight & Co Ltd St Albans

From Machine Intelligence 5

Edinburgh University Press

1970

# 22

# Shape Analysis by Use of Walsh Functions

N. H. Searle
Metamathematics Unit
University of Edinburgh

## INTRODUCTION

Walsh functions, and other sets of orthogonal boolean functions, are useful tools in the approximation of a given function. In the case of Walsh functions there are two particular advantages over other methods of function approximation. Firstly, the coefficients to be used in expressing the given function as a linear combination of the Walsh functions can be computed by a parallel algorithm. Secondly, where the given function represents a shape, or other object with spatial meaning, there is an interpretation of the approximation procedure under which the function is analysed into its spatial components. The former advantage is of particular importance where the amount of information to be processed is large, as it is when dealing with visual input to a computer.

An optical image is normally presented to a digital computer as an array, or retina, of regular cells. Associated with each cell is a digitized value, which may correspond to the intensity of the original image, say. This set of values is the *retinal function*. The cells may be square or hexagonal. The latter will in general give a better fit to the original image, but for conceptual reasons it is simpler to discuss the processing of retinal functions in terms of the former.

A *shape* is a binary-valued retinal function. For the purposes of analysing a shape in terms of its spatial structure, Walsh coefficients may also be regarded as binary-valued, positive coefficients taking the value 1, and negative coefficients the value 0, say.

In a pattern recognition system there are normally three stages: input, transformation, and discrimination. The transformation of the input serves the purpose of providing a description on the basis of which discrimination, or recognition, can take place. Complete systems of artificial pattern recognition can only hope to rival human performance when they are enhanced by

reasoning and linguistic capabilities of a high order. In the absence of sufficiently well-developed techniques in these areas, the present paper concerns itself solely with the first stage of the transformation process, and the place that Walsh analysis might have in the process.

Certain problems in the field of pattern recognition have already been tackled with considerable degrees of success. An obvious example is optical character recognition. This is a simple problem in the sense that the domain and range of the patterns are reasonably restricted, and that the recognition process takes place on a single character at a time. The complexity of, say, giving a description of a scene as viewed by a robot through a television camera eye is clearly much greater. In such a situation a general purpose pattern recognition system would suggest itself, rather than a collection of special purpose recognition systems. The methods described here are not special purpose, nor do they claim to be more than possibly a small contribution to a general purpose system.

There are circumstances under which recognition of visual input is not required. For example, a botanist may wish to have an accurate numerical description of the shape of a leaf, in order that he can compare it with other leaves grown under differing conditions. For such an application, he would not be interested in a specification of the type of leaf. The Walsh analysis initially transforms the input data into a new, more meaningful, set of numerical data, and this, rather than a complete recognition-type description, may be sufficient for, say, statistical work.

## WALSH FUNCTIONS

The Walsh functions are a complete set of orthogonal functions (Walsh 1923). They can be derived in several distinct ways, which give rise to more than one ordering of the functions. Two orderings are of particular interest, one for its computational properties, and the other for its interpretative properties. The first definition, for the interval [0, 1), can be extended to the whole or part of the real line and to $n$ dimensions, and is essentially the same as that originally proposed by Walsh.

*Definition* 1. The Walsh functions are $f_0, f_1, \ldots f_r, \ldots$

$$f_0(x) = 1, \qquad 0 \leqslant x < 1$$

$$\text{For } k \geqslant 1, f_k(x) = \begin{cases} f_l(2x), & 0 \leqslant x < \frac{1}{2} \\ (-1)^{k+l} f_l(2x-1), & \frac{1}{2} \leqslant x < 1 \end{cases}$$

where $l = \left[\frac{k}{2}\right]$.

It can be seen from the above definition that the Walsh functions are step-functions, which take the value $+1$ or $-1$ on each of the $2^n$ equal subintervals of $[0, 1) - \left[\frac{m}{2^n} \quad \frac{m+1}{2^n}\right), m = 0, 1, \ldots, 2^n - 1$ — where, for a given Walsh function, $f_k(x)$, $n$ is the smallest integer such that $k < 2^n$.

Therefore, the functions may be adequately represented by patterns of $+1$s and $-1$s.

*Example*

$$
\begin{array}{lrrrrrrrr}
f_0 = & 1 & & & & & & & \\
f_1 = & 1 & -1 & & & & & & \\
f_2 = & 1 & -1 & -1 & 1 & & & & \\
f_3 = & 1 & -1 & 1 & -1 & & & & \\
f_4 = & 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\
f_5 = & 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\
f_6 = & 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\
f_7 = & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1
\end{array}
$$

Note that each function is odd or even with respect to the midpoint of the interval and that the pattern $1\ 1\ -1\ -1$, say, represents the same function ($f_1$ in this case) as the pattern $1\ -1$; the difference being due to whether we consider $f_1$ to be one of the first four Walsh functions, or one of the first two.

The ordering of the functions which arises from this first definition of the functions will be referred to as the Walsh ordering, since it is identical to that produced by Walsh's own definition. It has the important property that the $k$th function, $f_k$, has exactly $k$ discontinuities on the interval $[0, 1)$.

If the patterns corresponding to the first $2^n$ Walsh functions are expanded so that they are all of length $2^n$ values, we can form the Walsh matrix, $W_n$. For example, when $n=3$, we obtain the following matrix, whose rows are the function values of the first eight Walsh functions:

$$
W_3 = \begin{pmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\
1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\
1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\
1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\
1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\
1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\
1 & -1 & 1 & -1 & 1 & -1 & 1 & -1
\end{pmatrix}
$$

It will be proved that $W_n$ is symmetrical for all positive integers $n$.

The next three definitions of the Walsh functions give orderings of the functions which are identical to one another. That they do so can be seen upon close inspection. In this new ordering the $k$th function does not in general have $k$ discontinuities on $[0, 1)$. Further, if $W'_n$ is the matrix of the Walsh function values in this new ordering, the $k$th row of $W'_n$ is not in general equal to the $k$th row of $W'_m$, where $m \neq n$, i.e., the $k$th function amongst the first $2^n$ Walsh functions is not also the $k$th amongst the first $2^m$ functions, in this ordering.

*Definition* 2. $W'_n$ is the matrix whose rows are the patterns of $+1$s and $-1$s which represent the first $2^n$ Walsh functions.

$$W'_0 = (1)$$

$$\text{For } n \geqslant 1,\ W'_n = \begin{bmatrix} W'_{n-1} & W'_{n-1} \\ W'_{n-1} & -W'_{n-1} \end{bmatrix}$$

This result is due to Lechner (1961).

These matrices are sometimes referred to as Hadamard matrices, but, as the notation used above suggests, here they will be considered to be a modified version of the Walsh matrices. The rows of $W'_n$ are a permutation of the rows of $W_n$, i.e., the first $2^n$ Walsh functions are the same set of functions under both definitions 1 and 2.

*Example*

$$W_2 = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{pmatrix} \qquad W'_2 = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}$$

*Definition* 3. The Rademacher functions are orthogonal boolean functions which are not complete in $L^2$. If $R_0(x)$, $R_1(x), \ldots R_m(x), \ldots$ are the Rademacher functions, then $R_n(x)$ is a step-function which takes the values $+1$ and $-1$ alternately in the $n+1$ equal sub-intervals of $[0, 1)$. They can be extended to other intervals and to $n$ dimensions.

$$R(x) = 1 \qquad 0 \leqslant x < 1$$

$$\text{For } n \geqslant 1.\ R_n(x) = \begin{cases} 1 & \dfrac{2k}{2^n} \leqslant x < \dfrac{2k+1}{2^n} \\ -1 & \dfrac{2k+1}{2^n} \leqslant x < \dfrac{2k+2}{2^n} \end{cases}$$

$$\text{for } k = 0, 1, \ldots, 2^n - 1.$$

As for the Walsh functions, the appropriate normalizing factor, $1/2^n$, has been omitted for clarity.

In fact, $R_n(x) = f_{2^n-1}(x)$ in the Walsh ordering; however, we are concerned here with the definition of the Walsh functions in terms of the Rademacher functions, rather than vice versa.

The Walsh functions can be defined as all possible products of Rademacher functions. No Rademacher function appears more than once in a give product, since $R_m(x) \,.\, R_m(x)$ is unity everywhere, for all $m$.

To determine which Rademacher functions are to be used in forming a given Walsh function, we write:

$$f_0(x) = R_0(x)$$

and $$f_{n_1 n_2 \ldots n_m}(x) = R_{n_1}(x) \,.\, R_{n_2}(x) \,.\, \ldots \,.\, R_{n_m}(x)$$

where $0 < n_1 < n_2 < \ldots < n_m$, and $f_{n_1 n_2 \ldots n_m}(x) = f_a(x)$,

where $a$ has unity in the $n_1, n_2, \ldots, n_m$th places after the binary point in its binary expansion and zero elsewhere.

It can be shown that

$$f_a(x) \,.\, f_b(x) = f_{a \oplus b}(x)$$

where $0 \oplus 1 = 1 \oplus 0 = 1$ and $1 \oplus 1 = 0 \oplus 0 = 0$;
and also that

$$f_a(x) = (-1)^{\{a,x\}}$$

where $\{a, x\}$ is the number of places in which both $a$ and $x$ have unity in their binary expansions.

It follows immediately that $f_a(x) = f_x(a)$ for all suitable $a$ and $x$.

Thus, for all non-negative integers $n$, $W'_n$ is symmetrical.

*Definition* 4. Suppose $p$ is a pattern of numbers; in particular, a pattern of $+1$s and $-1$s representing the values of a Walsh function as described above. Then we define the basic patterns

$$\bar{0} = 1 \quad 1$$

and

$$\bar{1} = 1 \quad -1$$

and the pattern products

$\bar{0} \,.\, p = p \quad p$ (i.e., the pattern $p$ followed by a repetition of itself)

and

$\bar{1} \,.\, p = p \quad -p$ (i.e., the pattern $p$ followed by the pattern in which each element is the negation of the corresponding element in $p$).

*Example*

$$\begin{array}{lllllllll} & & \bar{1} = 1 & -1 & & & & & \\ & \bar{0} \,.\, \bar{1} = 1 & -1 & 1 & -1 & & & & \\ \bar{1} \,.\, \bar{0} \,.\, \bar{1} = 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \end{array}$$

It is easily seen that the first $2^n$ Walsh functions can be expressed as all possible products of $n$ factors, each of which is a basic pattern. In other words, we may define a Walsh function as a basic pattern or the pattern product of a Walsh function and a basic pattern.

Note that $\bar{d}_1 \,.\, \bar{d}_2 \,.\, \ldots \,.\, \bar{d}_m \,.\, \bar{0} = \bar{d}_1 \,.\, \bar{d}_2 \,.\, \ldots \,.\, \bar{d}_m$.

The table below demonstrates the pattern product ordering of the first eight Walsh functions, covering the cases $n = 0, 1, 2, 3$.

| Walsh ordering × | .000 | .001 | .010 | .011 | .100 | .101 | .110 | .111 | Pattern product | New order |
|---|---|---|---|---|---|---|---|---|---|---|
| 000 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | $\bar{0}.\bar{0}.\bar{0}$ | 0 |
| 001 | 1 | 1 | 1 | 1 | −1 | −1 | −1 | −1 | $\bar{1}.\bar{0}.\bar{0}$ | 4 |
| 010 | 1 | 1 | −1 | −1 | −1 | −1 | 1 | 1 | $\bar{1}.\bar{1}.\bar{0}$ | 6 |
| 011 | 1 | 1 | −1 | −1 | 1 | 1 | −1 | −1 | $\bar{0}.\bar{1}.\bar{0}$ | 2 |
| 100 | 1 | −1 | −1 | 1 | 1 | −1 | −1 | 1 | $\bar{0}.\bar{1}.\bar{1}$ | 3 |
| 101 | 1 | −1 | −1 | 1 | −1 | 1 | 1 | −1 | $\bar{1}.\bar{1}.\bar{1}$ | 7 |
| 110 | 1 | −1 | 1 | −1 | −1 | 1 | −1 | 1 | $\bar{1}.\bar{0}.\bar{1}$ | 5 |
| 111 | 1 | −1 | 1 | −1 | 1 | −1 | 1 | −1 | $\bar{0}.\bar{0}.\bar{1}$ | 1 |

If we interpret the pattern product as a binary number, we obtain the new ordering, which is stated explicitly in the final column of the table. We shall refer to this as the pattern product ordering.

The concept of pattern products can be extended to $n$ dimensions, and as such can be used to define $n$-dimensional Walsh functions.

The pattern products in the penultimate column are closely related to the reflected binary code, and it is this relationship which gives the following result:

$f_{d_1d_2\ldots d_m}(x)$ in the pattern product ordering is

$f_{d_m.d_m\oplus d_{m-1}\ldots d_m\oplus d_{m-1}\oplus\ldots\oplus d_1}$ in the Walsh ordering.

A similar expression gives the inverse relationship.

It now follows that, in the Walsh ordering,

$$f_{d_1d_2\ldots d_m}(.\,r_1r_2\ldots r_m)=(-1)^{r_m\wedge d_1}f_{d_1\oplus d_2\;d_1\oplus d_3\ldots d_1\oplus d_m}(.\,r_1r_2\ldots r_m)$$

where $0\wedge 1=1\wedge 0=0\wedge 0=0$ and $1\wedge 1=1$.

If we continue to apply this transformation, we eventually obtain

$$f_{d_1d_2\ldots d_m}(.\,r_1r_2\ldots r_m)=$$
$$(-1)^{r_m\wedge d_1\oplus r_{m-1}\wedge d_1\oplus d_2\oplus\ldots\oplus r_2\wedge d_{m-2}\oplus d_{m-1}}f_{d_{m-1}\oplus d_m}(.\,r)$$

where $r=r_1r_2\ldots r_m$.

Since $f_0=1$ and $f_1=1\ \ -1$

$$f_{d_{m-1}\oplus d_m}(.\,r_1r_2\ldots r_m)=(-1)^{r_1\wedge d_{m-1}\oplus d_m}.$$

Hence,

$$f_{d_1d_2\ldots d_m}(.\,r_1r_2\ldots r_m)=(-1)^{r_m\wedge d_1\oplus r_{m-1}\wedge d_1\oplus d_2\oplus\ldots\oplus r_1\wedge d_{m-1}\oplus d_m}$$

and so

$$f_{d_1d_2\ldots d_m}(.\,r_1r_2\ldots r_m).f_{r_1r_2\ldots r_m}(.\,d_1d_2\ldots d_m)=1$$

i.e., $f_{d_1d_2\ldots d_m}(.\,r_1r_2\ldots r_m)=f_{r_1r_2\ldots r_m}(.\,d_1d_2\ldots d_m)$

i.e., $W_n$ is symmetrical for all non-negative integers $n$.

The two orderings of the Walsh functions both give matrices which are symmetrical for all $n \geqslant 0$.

The interpretation of the Walsh functions as pattern products leads to a result of great computational importance – the Fast Walsh Transform – which is described below.

## FUNCTION APPROXIMATION

Methods of function approximation which employ algebraic polynomials are not ideally suited to computer use, because of the large number of multiplications which are involved in the computation. The same criticism applies to the use of trigonometric functions, and it is only by approximating a given function with a linear combination of orthogonal boolean functions, preferably complete in $L^2$, that all multiplications can be avoided in the calculations.

Suppose $g(x)$ is an integrable function on $[0, 1)$. Then we can express $g$ as a linear combination of the Walsh functions, $f_0, f_1, \ldots, f_r, \ldots$

$$g(x) = \sum_{i=0}^{\infty} c_i \cdot f_i(x)$$

where $c_i$ is the $i$th Walsh coefficient, and is defined as the correlation between the given function, $g$, and the $i$th Walsh function, $f_i$:

$$c_i = \int_0^1 g(x) \cdot f_i(x)\, dx.$$

We can approximate a given function as closely as we please, since the partial sums,

$$s_N = \sum_{i=0}^{N} c_i \cdot f_i(x)$$

converge to the function, $g$, if $g$ is integrable square on $[0, 1]$.

Also, the coefficients, $c_i$, tend to zero as $i$ tends to infinity, since

$$\sum_{i=0}^{\infty} c_i^2 = \int_0^1 g^2(x)\, dx.$$

Suppose $g(x)$ is integrable on $[0, 1)$, and that we wish to approximate to $g$ by means of a linear combination of $2^n$ Walsh functions.

Then we first approximate to $g(x)$ by a step function $g_k$:

$$g_k = \frac{g_{\max} + g_{\min}}{2}, \quad k = 0, 1, \ldots, 2^n - 1$$

where $g_{\max}$ and $g_{\min}$ are the maximum and minimum values of $g(x)$ in the interval $\left[\frac{k}{2^n}, \frac{k+1}{2^n}\right)$.

Then we calculate the $2^n$ Walsh coefficients, $c_0, c_1, \ldots, c_{2^n-1}$, so that

$$g_k = \sum_{i=0}^{2^n-1} c_i \cdot f_i(x)$$

and $$c_i=\frac{1}{2^n}\sum_{k=0}^{2^n-1} g_k \cdot f_i(x), \quad \frac{k}{2^n}\leqslant x<\frac{k+1}{2^n}.$$

Since $f_i(x)$ always takes the value $+1$ or $-1$, for all $i$, the calculation of the Walsh coefficients only involves the addition or subtraction of the appropriate $g_k$; apart from the multiplication by the normalizing factor, $1/2^n$, which is not really a multiplication, but a shift, in any case.

## THE FAST WALSH TRANSFORM

Suppose we wish to evaluate the coefficient corresponding to the Walsh function whose pattern product representation is, say, $\bar{1} \cdot \bar{0} \cdot \bar{1}$. The coefficient required is

$$\tfrac{1}{8}(g_0-g_1+g_2-g_3-g_4+g_5-g_6+g_7)$$

where the $g_i$ are the function values, and the pluses and minuses correspond to the $+1$s and $-1$s of the pattern $\bar{1} \cdot \bar{0} \cdot \bar{1}$.

Now $$g_0-g_1+g_2-g_3-g_4+g_5-g_6+g_7$$
$$=(g_0-g_1)+(g_2-g_3)-(g_4-g_5)+(g_6-g_7)$$

which is a difference of sums of differences of $g_0, \ldots, g_7$, which we write as DSD – a D denoting a difference and an S a sum.

Note that there is a direct relationship between the Ds and the $\bar{1}$s of the pattern product and between the Ss and the $\bar{0}$s.

Hence we may compute all the $2^n$ coefficients as shown in the table below.

| For $n=3$ | Calculation of coefficients | | |
|---|---|---|---|
| Original data | Stage 1 | Stage 2 | Stage 3 |
| $g_0$ | $a_0=g_0+g_1$ | $b_0=a_0+a_2$ | $c_0=b_0+b_4$ |
| $g_1$ | $a_1=g_0-g_1$ | $b_1=a_1+a_3$ | $c_1=b_1+b_5$ |
| $g_2$ | $a_2=g_2+g_3$ | $b_2=a_0-a_2$ | $c_2=b_2+b_6$ |
| $g_3$ | $a_3=g_2-g_3$ | $b_3=a_1-a_3$ | $c_3=b_3+b_7$ |
| $g_4$ | $a_4=g_4+g_5$ | $b_4=a_4+a_6$ | $c_4=b_0-b_4$ |
| $g_5$ | $a_5=g_4-g_5$ | $b_5=a_5+a_7$ | $c_5=b_1-b_5$ |
| $g_6$ | $a_6=g_6+g_7$ | $b_6=a_4-a_6$ | $c_6=b_2-b_6$ |
| $g_7$ | $a_7=g_6-g_1$ | $b_7=a_5-a_7$ | $c_7=b_3-b_7$ |

Consider the fourth row: we first form $a_3$, the difference between $g_2$ and $g_3$; then the difference $a_1-a_3$, which is a difference of differences (since $a_1$ is the difference $g_0-g_1$); and finally the sum $b_3+b_7$, which is therefore the sum of differences of differences, since $b_3$ and $b_7$ are such.

Thus $c_3/2^n$ is a SDD and therefore is the value of the coefficient corresponding to the Walsh function $\bar{0} \cdot \bar{1} \cdot \bar{1}$ i.e., $f_4$ in the Walsh ordering.

To compute $2^n$ coefficients only $n \cdot 2^n$ additions and subtractions are required.

*Example*

| | | | |
|---|---|---|---|
| $g_0=0.2$ | $a_0=0.6$ | $b_0=2.0$ | $c_0=4.0$ |
| $g_1=0.4$ | $a_1=-0.2$ | $b_1=-0.4$ | $c_1=0.8$ |
| $g_2=0.6$ | $a_2=1.4$ | $b_2=-0.8$ | $c_2=-1.2$ |
| $g_3=0.8$ | $a_3=-0.2$ | $b_3=0$ | $c_3=0$ |
| $g_4=0.7$ | $a_4=0.8$ | $b_4=2.0$ | $c_4=0$ |
| $g_5=0.1$ | $a_5=0.6$ | $b_5=1.2$ | $c_5=-1.6$ |
| $g_6=0.9$ | $a_6=1.2$ | $b_6=-0.4$ | $c_6=-0.4$ |
| $g_7=0.3$ | $a_7=0.6$ | $b_7=0$ | $c_7=0$ |

and hence the coefficients in the Walsh ordering are

$$0.5 \quad 0 \quad -0.4 \quad -1.2 \quad 0 \quad 0 \quad -1.6 \quad 0.8$$

This method of computing the Walsh coefficients is closely analogous to the algorithm known as the Fast Fourier Transform (FFT), and for this reason is called the Fast Walsh Transform (FWT). It has elsewhere been called the Fast Walsh Fourier Transform and the Fast Hadamard Fourier Transform.

The FWT can be modified, as can the FFT, so that it becomes equivalent to the repeated execution of a much simpler algorithm (Pease 1968). The simplified algorithm is defined below, and each execution of it is equivalent to one stage of the FWT. It must be carried out $n$ times, therefore, to produce $2^n$ coefficients. (It is perhaps worth mentioning here that if it is executed $2n$ times, the coefficients produced after $n$ executions will be transformed back into the original data values; i.e., the FWT and its modified version are their own inverses.) Now that each stage of the modified FWT is identical, it is possible to think realistically (from an economic point of view) about constructing a special purpose machine to carry out each stage of the algorithm in parallel.

If the original data is $g_0, g_1, \ldots, g_1$, as above, then a single stage of the modified FWT is defined by:

| | | |
|---|---|---|
| $g'_0=g_0+g_1$ | | $g_0=g'_0$ |
| $g'_1=g_0-g_1$ | | $g_1=g'_2$ |
| $g'_2=g_2+g_3$ | | $g_2=g'_4$ |
| $g'_3=g_2-g_3$ | followed by | $g_3=g'_6$ |
| $g'_4=g_4+g_5$ | | $g_4=g'_1$ |
| $g'_5=g_4-g_5$ | | $g_5=g'_3$ |
| $g'_6=g_6+g_7$ | | $g_6=g'_5$ |
| $g'_7=g_6-g_7$ | | $g_7=g'_7$ |

or

$$
\begin{aligned}
g'_0 &= g_0 + g_1 \\
g'_1 &= g_2 + g_3 \\
g'_2 &= g_4 + g_5 \\
g'_3 &= g_6 + g_7 \quad \text{followed by} \quad g_i = g'_i, \quad \text{all } i. \\
g'_4 &= g_0 - g_1 \\
g'_5 &= g_2 - g_3 \\
g'_6 &= g_4 - g_5 \\
g'_7 &= g_6 - g_7.
\end{aligned}
$$

Although the FWT can be extended to $n$ dimensions, there is in fact no need to do so. An $n$-dimensional Walsh function is the product of $n$ one-dimensional Walsh functions; but such a product also defines another one-dimensional Walsh function in the sense of Definition 3. Hence, to each $n$-dimensional Walsh function there corresponds a one-dimensional Walsh function (the reverse is not true); and the computation of an $n$-dimensional coefficient (i.e., a coefficient corresponding to an $n$-dimensional function) can be executed by means of the (one-dimensional) FWT.

The coefficients produced by the FWT are in the pattern product ordering, and they must be permuted so as to be in the Walsh ordering, for reasons which are made clear below. The permutation is effected by means of a simple algorithm (see Appendix 2).

The algorithms of the FWT and the modified FWT are also in Appendix 2.

## SHAPES

A shape is defined as a bounded region which is the interior of a simple closed curve.

In a cellular array, each component of which takes the value 1 or 0 (black or white), a retinal shape is a set of connected cells of value 1.

A square cell has as neighbours the four cells which have common edges with it. A boundary corner is a corner of a cell of value 1, which is also a corner of a cell of value 0. A boundary cell has value 1 and at least two boundary corners.

The retinal function is the set of 1$s$ and 0$s$. We now consider a series of transformations upon such a function.

The coordinates of the boundary cells can be found by means of an edge-following algorithm (*see* Appendix 2) (Ledley 1965), or by a straightforward parallel computation which produces for a cell having value $C$, and neighbours having values $C_1$, $C_2$, $C_3$ and $C_4$, the value of the function

**if** $C=1$ **and** $C_1+C_2+C_3+C_4<4$ **then** 1 **else** 0 **close.**

The form of this function follows directly from the alternative definition of a boundary cell as one which has value 1 and at least one neighbour having value 0. In the case where all four neighbours have value 0 the central cell is a complete retinal shape in itself.

The centre of gravity of the two-dimensional lamina represented by the retinal function is easily found as the point whose coordinates are the averages of the coordinates of the cells forming the shape. The distances between the centre of gravity and the boundary corners is now calculated. Note that the boundary corners are equidistant along the perimeter of the retinal shape.

Let the distances from the centre of gravity to the boundary corners be $g_1, g_2, \ldots, g_m$. These are first approximated to by a set of values $g'_1, g'_2, \ldots, g'_N$, where $N$ is of the form $2^n$, and where

$$\frac{1}{N}\sum_{k=1}^{N} g'_k = \text{constant}.$$

The FWT can now be applied to this new set of function values to produce $N$ coefficients.

Since the point from which the measurements were taken (the centre of gravity) is relative to the retinal shape, the function values are translation independent, and so therefore are the resulting Walsh coefficients. Also, because the average value of $g'_k$ is constant, the transformation from the retinal shape to the coefficients is size independent.

However, the coefficients are rotation dependent, and so they do not prove a unique specification of the retinal shape. This problem can be partially overcome by using other sets of orthogonal boolean functions which are 'cyclical'.

For example:

$$\begin{aligned} f_0 &= \ 1 \quad\ 1 \quad -1 \quad\ 1 \\ f_1 &= \ 1 \quad\ 1 \quad\ 1 \quad -1 \\ f_2 &= -1 \quad\ 1 \quad\ 1 \quad\ 1 \\ f_3 &= \ 1 \quad -1 \quad\ 1 \quad\ 1 \end{aligned}$$

The typical function in such a set is a digital sequence with pseudonoise properties. Now the set of coefficients is unique. If a standard rotational position for a retinal shape is somehow fixed which gives coefficients

$$c_0 \quad c_1 \quad c_2 \quad c_3 \quad c_4 \quad c_5 \quad c_6 \quad c_7$$

then in general the same shape will produce coefficients

$$c_k \quad c_{k+1} \ldots c_7 \quad c_0 \quad c_1 \ldots c_{k-1}$$

It appears that such cyclical orthogonal boolean functions can be formed in complete sets of order $4m$ (Paley 1933), whereas Walsh functions can only be formed in complete sets of order $2^n$. However, Walsh functions have the important interpretative advantage of their resolution properties (below). Also, the Walsh functions are the only infinite, complete set of boolean orthogonal functions.

The transformation of the two-dimensional retinal shape into a one-dimensional set of function values (the distances between the centre of gravity and the boundary corners) has certain drawbacks, foremost of which is the

inability to deal with disconnected retinal 'shapes', in particular shapes with 'holes' in them. This problem could be overcome by standardizing the two-dimensional function with respect to translation, size, and rotation, and then applying the Walsh transform to the resulting retinal function of 1*s* and 0*s*. The computation involved is much greater than in the one-dimensional case, and we examine below how we can develop methods to deal with shapes with 'holes' on a one-dimensional basis; and in the process evolve a technique for processing retinal functions which represent a complex set of shapes.

There is another disadvantage of the centre-of-gravity/boundary corner method, which is that a certain amount of information is lost in the process of transforming the retinal function to the one-dimensional form. However, the information which is lost is of no importance for the purposes of spatial spectra analysis.

To reduce errors as much as possible, it is desirable that the number of boundary corners be close to a power of 2.

## RESOLUTION GRAPHS

The $k$th Walsh function in the Walsh ordering has exactly $k$ discontinuities on [0, 1); and the $k$th Walsh coefficient is therefore a measure of the correlation between a given function and a standard function with $k$ discontinuities. We can construct a spatial spectrum of a shape (a resolution graph) simply by plotting $c_m^2$ against $1/m$, for $m=1, 2, \ldots, 2^n-1$.

The Walsh coefficients can be used to determine whether the differences between shapes are due to, say, large-scale or small-scale features, and the resolution graph enables one to interpret the coefficients for such purposes at a glance.

Some shapes and their resolution graphs are shown in figure 1.

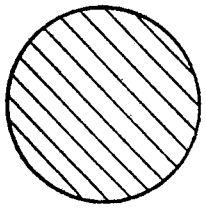
If the given function is produced from a shape in the manner described above, then a slight modification must be made in the construction of the resolution graph. In the Walsh functions, $f_{2m-1}$, $m=1, 2, \ldots$, there is an extra discontinuity between the initial and final values of the function. The spatial spectrum is formed by plotting

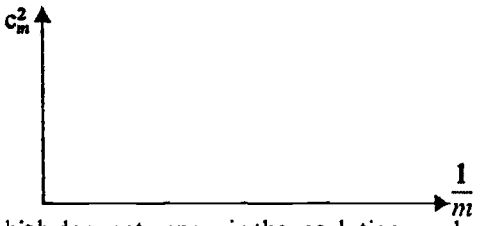
$$\frac{c_{2m}^2+c_{2m-1}^2}{2} \text{ against } \frac{1}{2m} \text{ for } m=1, 2, \ldots, \frac{N-2}{2}.$$

## PATTERNS OF SHAPES

The resolution graph of the retinal function shown in figure 2 would be extremely difficult to interpret, except to an experienced person, or where only a small amount of information was to be extracted. It is therefore desirable to consider figure 2 as a pattern of shapes; that is, as a black shape with two white shapes superimposed upon it. A tree which describes the topological structure of such a figure can be produced by Buneman's methods (*see* p. 383), although Buneman would regard the white areas as holes in the black shape.

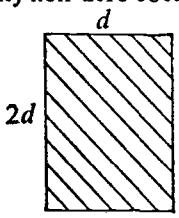
(a)

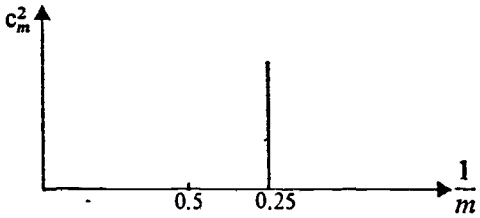





The only non-zero coefficient is $c_0$, which does not appear in the resolution graph.

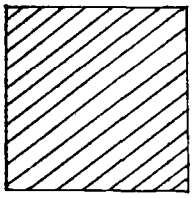
(b)

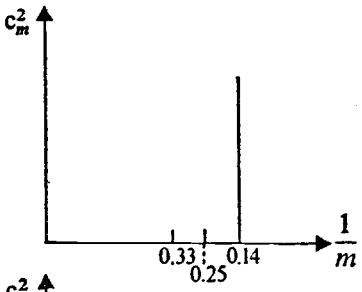


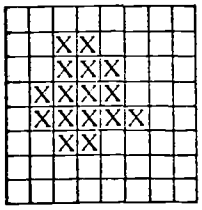

The height of the peak at 0.25 depends upon the size of the rectangle.

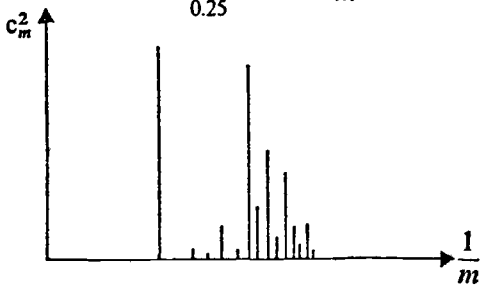
(c)



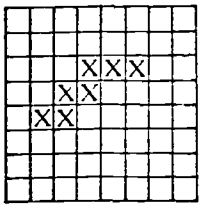



(d)






(e)



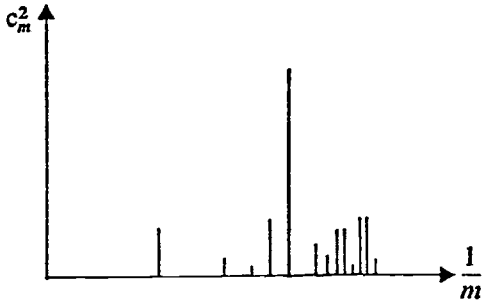



Figure 1

There are several methods, including one proposed by Buneman, for producing a set of retinal functions, each of which represents a shape, from a retinal function which represents a pattern of shapes. Unfortunately, this task of decomposition cannot be carried out by means of a parallel algorithm (Minsky and Papert 1969).

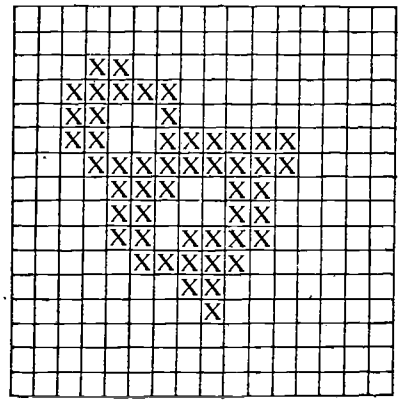


Figure 2. The x*s* denote black squares

The specification of a retinal function such as figure 2 consists, therefore, of a terminal string, produced by Buneman's grammar, with a set of Walsh coefficients attached to each node of the corresponding tree, i.e., as a combination of the sets of Walsh coefficients.

**Acknowledgements**

This work was supported by a grant from the Science Research Council. I am extremely grateful to many people, especially Dr B. Meltzer, for their suggestions and interest.

## REFERENCES

Lechner, R. (1961) A transform theory for functions of binary variables. Theory of switching Harvard Computation Lab., Cambridge, Mass. Progress Report No. BL-30, Section X, 1–37.

Ledley, R.S. (1965) *Use of computers in biology and medicine*, p. 340. New York: McGraw-Hill.

Minsky, M. & Papert, S. (1969) *Perceptrons*, p. 74. Cambridge, Mass.: MIT Press.

Paley, R.E.A.C. (1933) On orthogonal matrices. *J. Math. Phys.*, **12**, 311–20.

Pease, M.C. (1968) An adaptation of the fast fourier transform for parallel processing. *J. Ass. comput. Mach.*, **15**, 252–64.

Polyak, B.T. & Shreider, Yu.A. (1966) The application of Walsh functions in approximate calculations. *Problems in theory of mathematical machines: collection II.* pp. 174–90. Moscow: Fizmatgiz.

Walsh, J.L. (1923) A closed set of orthogonal functions. *Amer. J. Math.*, **55**, 5–24.

## APPENDIX 1

The original retinal function is binary-valued and the main reason for not working with the data in this form is the increased complexity of the two-dimensional calculations which would be involved. Even in one dimension the most convenient form in which to represent the outline of a retinal shape would be by a pair of ternary-valued functions, one recording whether in moving from one boundary cell to the next the $x$-coordinate increases, decreases, or stays unaltered, the other giving the corresponding information about the $y$-coordinates.

It would appear to be useful, therefore, if special methods could be found for dealing with functions which take only a finite number of different values, and in particular binary and ternary functions. There is a reasonable hope that this might be achieved.

## APPENDIX 2

### 1. Permutation of $2^n$ Walsh coefficients from pattern product ordering to Walsh ordering.

```
function conv n n; vars i k;
logand(logbits(j), 1)→k;
for i, step (1, 1, n−1) do
if logand(logbits(logand(j, logshift(n, 1−i))), 1)=0 then goto back close;
k+logshift(1, i)→k;
back: repeat;
k
end;
```

note: *logbits*($j$) is a function whose value is the number of bits which are 1 in the binary expansion of $j$.

### 2. Fast Walsh Transform

```
function fwt n; vars i j k l p q w;
0→k;
back logshift:(1, k)→1;
for i, step (0, 21, 2**n−21) with j, step (0, 1, l−1) do
w(i+j)→p;
w(i+j+1)→q;
p+q→w(i+j);
p−q→w(i+j+1);
repeat;
k+1→k;
if not(k=n) then goto back close;
end;
```

### 3. Modified Fast Walsh Transform

```
function walsh n; vars i j h g a k m;
0→i;
2**n→m;
```

```
back: for j, step(0, 2, (m-1)/2) do
logshift(j, -1)→k;
g(k)→h(j)+h(j+1);
g(k+m/2)→h(j)-h(j+1);
repeat;
h→j;
g→h;
j→g;
i+1-i;
if not(i=n) then goto back close
end;
```

**4. Edge follower**

```
function follow; vars startx starty k a c;
read( startx starty);
0→k;
startx→i;
starty→j;
1: check;
move(i, j-1, 2); move(i-1, j 4); move(i, j+1, 7); goto 8;
10: check;
move(i+1, j, 8); move(i, j-1, 2); move(i-1, j, 4); goto 7;
100: check;
move(i-1, j, 4); move (i, j+1, 7); move(i+1, j, 8); goto 2;
200: check;
move(i, j+1, 7); move(i+1, j, 8); move(i, j-1, 2); goto 4;
2: if not(c(i+1, j-1)=1) then goto 3; i+1→i; 3: j-1→j; goto 10;
4: if not(c(i-1, j-1)=1) then goto 6; j-1→j; 6: i-1→i; goto 1;
7: if not(c(i-1, j+1)=1) then goto 9; i-1→i; 9: j+1→j; goto 100;
8: if not(c(i+1, j+1)=1) then goto 11; j+1→j; 11: i+1→i; goto 200;
30: stop;
end;
function check; vars i j c startx starty k;
if i=startx and j=starty and not (k=2) then goto 30 close
end;
function move i j x;
if c(i, j)=1 then goto x close
end;
```

The above functions have been very freely translated from the original language.

# APPLICATIONS OF WALSH FUNCTIONS

## 1970 PROCEEDINGS

## *SYMPOSIUM AND WORKSHOP*

---

*Sponsors* | Naval Research Laboratory and University of Maryland

# A LOGICAL WALSH-FOURIER TRANSFORM

N.H.Searle, Metamathematics Unit,
University of Edinburgh, Scotland

## Introduction

The main advantage of the fast Walsh-Fourier transform over the fast Fourier transform is computational. The Walsh functions are essentially binary-valued, and hence the transform involves no multiplications. Suppose the transform is being applied to a set of binary data, then there are two respects in which it appears that useful modifications could be made.

Firstly, the transform produces a set of Fourier coefficients which contains no more information than the data, but some members of which contain more bits than a single datum.

Example

| | | | | |
|---|---|---|---|---|
| Data | 11 | 10 | 00 | 11 |
| Coeffs. | 1000 | -10 | 10 | 100 |

If each datum occupies one computer word, the coefficients may require more space, unless the redundancy can be removed.

Secondly, both the Walsh functions and the data are of a binary nature, yet the computation is arithmetic rather than Boolean, which, intuitively, would be more appropriate.

It will be shown how the redundancy can be removed, and also how the transform might be made non-arithmetic.

## Redundancy

The data is taken to consist of $N(=2^n)$ bits, from which N Fourier coefficients are to be computed by means of the fast Walsh-Fourier transform. Clearly, if there is to be no redundancy, each coefficient must take either of only two possible values i.e. the set of Fourier coefficients will contain only N bits.

The obvious choice for the set of values which the coefficients may take is 0 and 1. In the case, the transform from data to coefficients will constitute a mapping of the first $2^N$ binary numbers into, and in fact onto, themselves.

It will be shown that all binary data can be reduced to the form where each datum consists of a single bit.

There is one further restriction which must be placed on the data. This is, that the first datum must equal unity. The reason for this condition is that the Walsh functions each take the value +1 as frequently as they take the value -1, except for the first Walsh function, which has the value +1 everywhere. Therefore, there exists an imbalance between positive and negative values which must be compensated for here.

## Modified Transform

The Fourier coefficients are computed in the normal manner by the fast Walsh-Fourier transform. They are then reduced to the values 1 and 0, depending upon whether the original value is positive or non-positive, respectively.

Example

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Data | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 |
| Coeffs. | 5 | -1 | -1 | 1 | 1 | 3 | -1 | 1 |
| Reduced to | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 |

Note that the first Fourier coefficient will always be positive, except when each datum is zero (which is not allowed because of the restriction placed on the form of the data), and hence the first " binary" coefficient will always be unity.

The important property of the fast Walsh-Fourier transform that it is its own inverse is preserved under this modification.

Example

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Data | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |
| Fourier coeffs. | 4 | 0 | 2 | 2 | 0 | 0 | 2 | -2 |
| Binary coeffs. | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 |

If the binary coefficients are taken as data and the transform is applied again, we obtain

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Fourier coeffs. | 4 | 2 | -2 | 0 | 2 | 0 | 0 | 2 |
| Binary coeffs. | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |

The reason that a Fourier coefficient equal to zero is reduced to a binary coefficient of zero, rather than unity, is again that there is an imbalance between the positive and negative values of the Walsh functions.

## Restrictions

(i) If the first of the N data bits is zero, then the procedure is as follows:

The data is complemented; the

Fourier coefficients, and then the binary coefficients, are computed: the binary coefficients are then complemented.

In this case, the first binary coefficient will always be zero.

Example

| | | | | |
|---|---|---|---|---|
| Data | 0 | 1 | 1 | 1 |
| Complemented | 1 | 0 | 0 | 0 |
| Binary coeffs. | 1 | 1 | 1 | 1 |
| Complemented | 0 | 0 | 0 | 0 |

(ii) If each of the N data contains m bits rather than just one, then the procedure is as follows:

The modified fast Walsh-Fourier transform which produces binary coefficients is applied to m sets of data, each of which contains N bits; the jth set consists of the jth bit from each of the N m-bit data. In each case, the binary coefficients replace, positionally, the data from which they were derived.

Example

| | | | | |
|---|---|---|---|---|
| Data | 101 | 101 | 100 | 010 |
| | 1 | 1 | 1 | 0 |
| | 0 | 0 | 0 | 1 |
| | 1 | 1 | 0 | 0 |
| Binary coeffs. | 1 | 1 | 1 | 0 |
| | 0 | 1 | 1 | 1 |
| | 1 | 0 | 1 | 0 |
| | 101 | 110 | 111 | 010 |

(iii) Some applications of Walsh functions, particularly in the field of pattern recognition, involve the use of a two- or even higher-dimensional fast Walsh-Fourier transform.

In general, the n-dimensional transform can be reduced quite simply to the one-dimensional case

Suppose the array below represents the values of a two-dimensional binary function.

| | | | |
|---|---|---|---|
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 |

Then the two-dimensional transform will produce the same Fourier, and therefore binary, coefficients as the one-dimensional transform applied to

1 0 1 1 1 1 0 0 0 1 0 1 1 0 0 1

i.e. the set of data produced if the rows of the array are written out one after the other.

The same would be true if the columns were written one after another, since to each two-dimensional Walsh function with indices i, j there corresponds another with indices j,i.

The mapping from the indices of the resultant two-dimensional binary coefficient to the index of the appropriate one-dimenaional binary coefficient depends, of course, upon whether the array is read by rows or columns.

The reason for the into and onto mapping of two- (and higher-) dimensional binary coefficients is that a two-dimensional Walsh function is a product of two one-dimensional Walsh functions, which can be multiplied together in another sense to produce another one-dimensional function.

Example

| | | | | |
|---|---|---|---|---|
| Walsh | +1 | +1 | -1 | -1 |
| functions | +1 | -1 | -1 | +1 |

Multiplied to produce a two-dimensional Walsh function:

| | | | |
|---|---|---|---|
| +1 | +1 | -1 | -1 |
| -1 | -1 | +1 | +1 |
| -1 | -1 | +1 | +1 |
| +1 | +1 | -1 | -1 |

or

| | | | |
|---|---|---|---|
| +1 | -1 | -1 | +1 |
| +1 | -1 | -1 | +1 |
| -1 | +1 | +1 | -1 |
| -1 | +1 | +1 | -1 |

Multiplied to produce another one-dimensional function:

+1 +1 -1 -1 -1 -1 +1 +1 -1 -1 +1 +1 +1 +1 -1 -1

or

+1 -1 -1 +1 +1 -1 -1 +1 -1 +1 +1 -1 -1 +1 +1 -1

The method of storing an array in a computer shows an analagous similarity between n-dimensional and one-dimensional cases.

## A Logical Transform

We now have a modified fast Walsh-Fourier transform (based on the binary-valued Walsh functions) operating upon binary data to produce binary Fourier coefficients.

However, the computation is arithmetic

producing non-binary Fourier coefficients before reduction according to the positive-non-positive criterion.

The fundamental operations in the fast Walsh-Fourier transform are addition and subtraction of two data. An attempt to find two binary operators, which could be applied to a pair of bits to produce a binary result, and which could replace addition and subtraction, has proved impossible if the more desirable properties of the arithmetic fast Walsh-Fourier transform are to be preserved.

Corresponding to the criterion of whether a Fourier coefficient is positive or not, there is a boolean threshold function. The coefficients can be produced by means of a set of such functions, so that, not only do the final set of coefficients contain no redundant information, but no redundant information is generated in the process of the transform.

Suppose we have four data bits, $X_1$, $X_2$, $X_3$, $X_4$. Then the binary coefficient corresponding to the Walsh function which takes the value +1 in the first half of the interval and -1 in the second half is

$$\begin{aligned} &\bar{X}_2 \cdot \bar{X}_3 \cdot \bar{X}_4 \\ +\ &X_2 \cdot X_3 \cdot \bar{X}_4 \\ +\ &X_2 \cdot \bar{X}_3 \cdot X_4 \\ +\ &X_2 \cdot \bar{X}_3 \cdot \bar{X}_4 \end{aligned}$$

where '-', '.' and ' + ' denote boolean negation, conjunction and disjunction, respectively. It is assumed that $X_1$ is unity.

If the negated data are replaced by 0 and the remainder by 1, then the binary numbers from 0 to 3, or their complements, are obtained from the disjuncts of the above function. This is true when there are N data bits, and for all the functions corresponding to the N coefficients.

Suppose the data is 1101. Then we can determine the value of each binary coefficient by discovering whether 101 ($X_2 \cdot \bar{X}_3 \cdot X_4$) appears as a disjunct in each function as $X_2 \cdot \bar{X}_3 \cdot X_4$ or $\bar{X}_2 \cdot X_3 \cdot \bar{X}_4$.

A disjunct has the value 1 if it contains $X_2$, and 0 if it contains $\bar{X}_2$.

We can also represent the Walsh functions by 1s and 0s, merely replacing the naturally-occurring -1s by 0s.

Consider the table below:

| Walsh fns. | 1100 | 1001 | 1010 |
|---|---|---|---|
| Coeff. functions | $\bar{X}_2 \cdot \bar{X}_3 \cdot \bar{X}_4$ | $X_2 \cdot \bar{X}_3 \cdot X_4$ | $X_2 \cdot X_3 \cdot \bar{X}_4$ |
| | $+ X_2 \cdot X_3 \cdot \bar{X}_4$ | $+ \bar{X}_2 \cdot X_3 \cdot X_4$ | $+ \bar{X}_2 \cdot \bar{X}_3 \cdot \bar{X}_4$ |
| | $+ X_2 \cdot \bar{X}_3 \cdot X_4$ | $+ \bar{X}_2 \cdot \bar{X}_3 \cdot \bar{X}_4$ | $+ \bar{X}_2 \cdot X_3 \cdot X_4$ |
| | $+ X_2 \cdot \bar{X}_3 \cdot X_4$ | $+ \bar{X}_2 \cdot \bar{X}_3 \cdot X_4$ | $+ \bar{X}_2 \cdot X_3 \cdot \bar{X}_4$ |

If the data is X and the jth value of the ith Walsh function is W(i,j), then the coefficients are given by

$$C_{2K} = EQ(W(2K,2), EQ(X_2, D_{NQ(X;NQ(W(1),W(2K)))}))$$

where EQ gives the result 1 when both arguments are 1, or both are 0, and 0 otherwise

NQ is bit-by-bit non-equivalence.
W(J) is the Jth Walsh function.
$D_I$ is the Ith disjunct of the boolean function corresponding to W(1).

Since $W(1).W(2N) = W(2N+1)$, the above expression can be simplified:

$$C_{2K} = EQ\left[W(2K,2), EQ(X_2, D_{NQ[X,\overline{W_{2K+1}}]})\right]$$

$$= EQ\left[W(2K,2), EQ(X_2, D_{EQ[X,W_{2K+1}]})\right]$$

where EQ is interpreted as bit-by-bit equivalence, where appropriate.

Also,

$$C_{2K+1} = EQ\left[W(2K+1,2), EQ(X_2, D_{EQ(X,W_{2K})})\right]$$

and $W(I,2) = 1$ iff $I < N/2$.

Example

| | | | |
|---|---|---|---|
| W(0) | 1 | 1 | 1 |
| W(3) | 0 | 1 | 0 |
| W(2) | 0 | 0 | 1 |

Disjuncts of W(1)

| | |
|---|---|
| $D_0$ | 0 |
| $D_1$ | 1 |
| $D_2$ | 1 |
| $D_3$ | 1 |

(i) Data: 1 1 0 0

The array becomes:

| | | |
|---|---|---|
| 1 | 0 | 0 |
| 0 | 0 | 1 |
| 0 | 1 | 0 |

(The second and third columns are complemented, as a result of the non-equivalence with $X_3$ and $X_4$.)

The rows of the array, complemented if necessary, indicate which disjuncts give the values of the coefficients.

| $D_3$ | $D_1$ | $D_2$ |
|---|---|---|
| 1 | 1 | 1 |

The last two are complemented, since $W(2,2) = W(3,2) = 0$.

Thus the binary coefficients are

1 1 0 0

(ii) Data: 1 0 1 0

The array becomes:

| | | |
|---|---|---|
| 0 | 1 | 0 |
| 1 | 1 | 1 |
| 1 | 0 | 0 |

| $D_1$ | $D_0$ | $D_3$ |
|---|---|---|
| 1 | 0 | 1 |

These are all complemented, since $X_2 = 0$ ; and the second and third are again complemented, as in (i).

Hence, the coefficients are

1 0 0 1

## The Fast Haar-Fourier Transform

The transform based on the Haar functions does not have the interpretative properties of the fast Walsh-Fourier Transform, but it is considerably faster. The above methods can be modified for application with the Haar transform.

## References


1. Walsh, J.L. 'A Closed Set of Normal Orthogonal Functions', 1923, Amer.J. Math., 45, 5-24

2. Lechner,R. ' A Transform Theory for Functions of Binary Variables', in 'Theory of Switching', Harvard Computatic Lab. Cambridge, Mass., Progress Rept. No.BL - 30, Sec. X, 1-37, November 1961.

3. Polyak, B.T., and Shreider, Yu.A., 'The Application of Walsh Functions in Approximate Calculations', 1966, in 'Problems in Theory of Mathematical Machines: Collection II', 174-190, Fizmatgiz, Moscow.

4. Pease, M.C., 'An Adaption of the Fast Fourier Transform for Parallel Processing', J.ACM, 15, 2 (April 1968), 252-264.

5. Harmuth, H.F., 'Transmission of Information by Orthogonal Functions', Springer-Verlag (Berlin, Heidelberg, New York) 1969.

6. Sheng, C.L. , 'Threshold Logic', Academic Press (London, New York) 1969.


* * * * * * * * * * * * * * * * * * *